

**Proceedings of the 15th International Conference on
Educational Data Mining**

Antonija Mitrovic & Nigel Bosch (eds).

24–27 July, 2022
University of Durham
Durham, United Kingdom



©2022 International Educational Data Mining Society

This work is licensed under the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-nc-nd/4.0/> or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.

Download copies of this and other EDM proceedings from:

International Educational Data Mining Society (IEDMS)

<https://educationaldatamining.org>

Proceedings of the 15th International Conference on Educational Data Mining.

Antonija Mitrovic & Nigel Bosch (eds).

July 24–27, 2022. Durham, United Kingdom.

ISBN 978-1-7336736-3-1

Preface

For this 15th iteration of the International Conference on Educational Data Mining (EDM 2022), the conference returned to England, this time to Durham, with an online hybrid format for virtual participation as well. EDM is organized under the auspices of the International Educational Data Mining Society. The conference, held July 24th through 27th, 2022, follows fourteen previous editions (fully online in 2021 and 2020, Montréal 2019, Buffalo 2018, Wuhan 2017, Raleigh 2016, Madrid 2015, London 2014, Memphis 2013, Chania 2012, Eindhoven 2011, Pittsburgh 2010, Cordoba, 2009 and Montréal 2008).

The theme of this year's conference is Inclusion, Diversity, Equity, and Accessibility (IDEA) in EDM Research and Practice. This theme emphasizes the importance of considering and broadening who is included – or not included – in EDM, and why. Furthermore, the theme speaks to the importance of IDEA considerations in all stages of the research process, from participant recruitment and selection, data collection, methods, analysis, results, to the application of research results in the future. The conference features three invited talks: Jennifer Hill, Professor of Applied Statistics at New York University, USA; René Kizilcec, Assistant Professor of Information Science at Cornell University, USA; and Judy Robertson, Professor of Digital Learning at the University of Edinburgh, Scotland. As in the past few years of EDM, this year's conference also includes an invited keynote talk by the 2021 winner of the EDM Test of Time Award. The talk is delivered by Tiffany Barnes, Distinguished Professor of Computer Science at North Carolina State University, USA.

This year's EDM conference continued the double-blind review process that started in 2019. The program committee was once again extended, this time using an interest survey process, to better reflect the community presenting works and to keep the review load for each member manageable. EDM received 90 submissions to the full papers track (10 pages), of which 26 were accepted (28.9%), while a further 12 were accepted as short papers (6 pages) and 14 as posters (4 pages). There were 56 submissions to the short paper track, of which 17 were accepted (30.4%) and a further 20 were accepted as posters. The poster and demo track itself accepted 10 contributions out of 20 submissions.

The EDM 2022 conference also held a Journal of Educational Data Mining (JEDM) Track that provides researchers a venue to deliver more substantial mature work than is possible in a conference proceeding and to present their work to a live audience. The papers submitted to this track followed the JEDM peer review process. Five papers were submitted and two papers are featured in the conference's program.

The main conference invited contributions to an Industry Track in addition to the main track. The EDM 2022 Industry Track received six submissions of which four were accepted. The EDM conference also continues its tradition of providing opportunities for young researchers to present their work and receive feedback from their peers and senior researchers. The doctoral consortium this year features nine such presentations.

In addition to the main program, there are six workshops and tutorials: *Causal Inference in Edu-*

ational Data Mining (Third Annual Half-Day Workshop), 6th Educational Data Mining in Computer Science Education (CSEDM) Workshop, FATED 2022: Fairness, Accountability, and Transparency in Educational Data, The Third Workshop of The Learner Data Institute: Big Data, Research Challenges, & Science Convergence in Educational Data Science, Rethinking Accessibility: Applications in Educational Data Mining, and Tutorial: Using the Open Science Framework to promote Open Science in Education Research.

We thank the sponsors of EDM 2022 for their generous support: DuoLingo, ETS, Durham University Department of Computer Science, and Durham University School of Education. We are also thankful to the senior program committee and regular program committee members and reviewers, without whose expert input this conference would not be possible. Finally, we thank the entire organizing team and all authors who submitted their work to EDM 2022. And we thank EasyChair for their infrastructural support.

<i>Antonija Mitrovic</i>	University of Canterbury	Program Chair
<i>Nigel Bosch</i>	University of Illinois Urbana–Champaign	Program Chair
<i>Alexandra I. Cristea</i>	Durham University	General Chair
<i>Chris Brown</i>	Durham University	General Chair

July 23rd, 2022
Durham, England, UK

Organizing Committee

General Chairs

- Alexandra I. Cristea (Durham University, UK)
- Chris Brown (Durham University, UK)

Program Chairs

- Antonija Mitrovic (University of Canterbury, NZ)
- Nigel Bosch (University of Illinois Urbana–Champaign, US)

Workshop & Tutorial Chairs

- Angela Stewart (Carnegie Mellon University, US)
- Steven Bradley (Durham University, UK)

Industry Track Chairs

- Carol Forsyth (Educational Testing Service, US)
- Stephen Fancsali (Carnegie Learning, Inc., US)

Doctoral Consortium Chairs

- Neil Heffernan (Worcester Polytechnic Institute, US)
- Craig Stewart (Durham University, UK)
- Armando Toda (Durham University, UK)
- Carol Forsyth (Educational Testing Service, US)

JEDM Track Chairs

- Sharon Hsiao (Santa Clara University, US)
- Luc Paquette (University of Illinois Urbana–Champaign, US)

Poster & Demo Track Chairs

- Frederick Li (Durham University, UK)
- Michelle P. Banawan (Arizona State University, US)
- Hassan Khosravi (University of Queensland, AU)

Publication/Proceedings Chairs

- Andrew M. Olney (University of Memphis, US)
- Tahir Aduragba (Durham University, UK)

Accessibility Chairs

- JooYoung Seo (University of Illinois Urbana–Champaign, US)
- Paul Salvador Inventado (California State University Fullerton, US)

Diversity, Equity, and Inclusion Chair

- Agathe Merceron (Beuth University of Applied Sciences, DE)

Online/Hybrid Experience Chairs

- LuEttaMae Lawrence (University of California Irvine, US)
- Stephen Hutt (University of Pennsylvania, US)

Publicity and Sponsorship Chair

- Effe Law (Durham University, UK)

Web Chair

- Lei Shi (Durham University, UK)

Local Organisers

- Jim Ridgeway
- Peter Tymms
- Suncica Hadzidedic
- Stamos Katsigiannis
- Elaine Halliday
- Georgina Sales
- Judith Williams
- Jingyun Wang
- Dorothy Monekosso
- Nelly Bencomo
- Jindi Wang

IEDMS Officers

Tiffany Barnes, President North Carolina State University, USA
Mingyu Feng, Treasurer WestEd, USA

IEDMS Board of Directors

Rakesh Agrawal Data Insights Laboratories, USA
Ryan Baker University of Pennsylvania, USA
Michel Desmarais Polytechnique Montréal, Canada
Neil Heffernan Worcester Polytechnic Institute, USA
Kenneth Koedinger Carnegie Mellon University, USA
Luc Paquette University of Illinois Urbana–Champaign, USA
Anna Rafferty Carleton College, USA
Mykola Pechenizkiy Eindhoven University of Technology, Netherlands
Kalina Yacef University of Sydney, Australia

Senior Program Committee

Agathe Merceron Beuth University of Applied Sciences Berlin
Alex Bowers Columbia University
Andrew Lan University of Massachusetts at Amherst
Andrew M. Olney University of Memphis
Anna Rafferty Carleton College
Caitlin Mills University of New Hampshire
Collin Lynch North Carolina State University
Cristobal Romero Department of Computer Sciences and Numerical Analysis
Dragan Gasevic Monash University
Gautam Biswas Vanderbilt University
Irena Koprinska The University of Sydney
James Lester North Carolina State University
Jesus G. Boticario UNED
Jill-Jênn Vie Inria
John Stamper Carnegie Mellon University
Jonathan Rowe North Carolina State University
José González-Brenes Chegg
Justin Reich Massachusetts Institute of Technology
Kasia Muldner Carleton University

Kristy Elizabeth Boyer	University of Florida
Luc Paquette	University of Illinois Urbana–Champaign
Martina Rau	University of Wisconsin - Madison
Michel Desmarais	Polytechnique Montréal
Min Chi	BeiKaZhouLi
Mingyu Feng	WestEd
Neil Heffernan	Worcester Polytechnic Institute
Niels Pinkwart	Humboldt-Universität zu Berlin
Noboru Matsuda	North Carolina State University
Philip I. Pavlik Jr.	University of Memphis
Radek Pelánek	Masaryk University Brno
Roger Azevedo	University of Central Florida
Ryan Baker	University of Pennsylvania
Sebastián Ventura	University of Cordoba
Shaghayegh Sahebi	University at Albany - SUNY
Sidney D’Mello	University of Colorado Boulder
Stefan Trausan-Matu	University Politehnica of Bucharest
Stephan Weibelzahl	Private University of Applied Sciences Göttingen
Stephen Fancsali	Carnegie Learning, Inc.
Steven Ritter	Carnegie Learning, Inc.
Vanda Luengo	Sorbonne Université - LIP6
Vincent Alevén	Carnegie Mellon University
Zach Pardos	University of California, Berkeley

Program Committee

Abhinava Barthakur	University of South Australia
Aditi Mallavarapu	University of Illinois Chicago
Ahmad Mel	Ghent University
Ali Darvishi	The University of Queensland
Amal Zouaq	Polytechnique Montréal
Amelia Zafra Gómez	Department of Computer Sciences and Numerical Analysis
Anis Bey	Université Paul Sabatier
Anna Finamore	Universidade Lusófona
Anthony F. Botelho	Worcester Polytechnic Institute
April Murphy	Carnegie Learning, Inc.
Aïcha Bakki	Le Mans University
Beverly Park Woolf	University of Massachusetts at Amherst
Bitá Akram	North Carolina State University

Buket Doğan	Marmara Üniversitesi
Carol Forsyth	Educational Testing Service
Chris Piech	Stanford University
Clara Belitz	University of Illinois Urbana–Champaign
Claudia Antunes	Instituto Superior Técnico - Universidade de Lisboa
Costin Badica	University of Craiova
Craig Zilles	University of Illinois Urbana–Champaign
Cynthia D’Angelo	University of Illinois Urbana–Champaign
David Pritchard	Massachusetts Institute of Technology
Destiny Williams-Dobosz	University of Illinois Urbana–Champaign
Diego Zapata-Rivera	Educational Testing Service
Donatella Merlini	Università di Firenze
Ean Teng Khor	Nanyang Technological University
Eliana Scheihing	Universidad Austral de Chile
Ella Haig	School of Computing, University of Portsmouth
Emily Jensen	University of Colorado Boulder
Erik Hemberg	ALFA
Feifei Han	Griffith University
Frank Stinar	University of Illinois Urbana–Champaign
Giora Alexandron	Weizmann Institute of Science
Guanliang Chen	Monash University
Guojing Zhou	University of Colorado Boulder
Hannah Valdiviejas	University of Illinois Urbana–Champaign
Hassan Khosravi	The University of Queensland
Hatim Lahza	The University of Queensland
Howard Everson	SRI International
Ivan Luković	University of Novi Sad
Jeremiah Folsom-Kovarik	Soar Technology, Inc.
Jia Zhu	Florida International University
Jiangang Hao	Educational Testing Service
Jihyun Park	Apple, Inc.
Jina Kang	University of Illinois Urbana–Champaign
JooYoung Seo	University of Illinois Urbana–Champaign
Jose Azevedo	Instituto Politécnico do Porto
José Raúl Romero	University of Cordoba
Joshua Gardner	University of Washington
Juho Leinonen	University of Helsinki
Julien Broisin	University of Toulouse
Julio Guerra	University of Pittsburgh
Jun-Ming Su	National University of Tainan
Keith Brawner	United States Army Research Laboratory

Khushboo Thaker	University of Pittsburgh
Lan Jiang	University of Illinois Urbana–Champaign
Ling Tan	Australian Council for Educational Research
LuEttaMae Lawrence	University of California Irvine
Mar Perez-Sanagustin	Pontificia Universidad Católica de Chile
Marcus Specht	Delft University of Technology
Marian Cristian Mihaescu	University of Craiova
Martin Hlosta	Swiss Distance University of Applied Sciences
Matt Myers	University of Delaware
Mehmet Celepkolu	University of Florida
Michelle Banawan	Arizona State University
Mirko Marras	École Polytechnique Fédérale de Lausanne (EPFL)
Nathan Henderson	North Carolina State University
Nathaniel Blanchard	Colorado State University
Nicholas Diana	Colgate University
Olga C. Santos	aDeNu Research Group (UNED)
Patrick Donnelly	Oregon State University Cascades
Paul Hur	University of Illinois Urbana–Champaign
Paul Salvador Inventado	California State University Fullerton
Paul Stefan Popescu	University of Craiova
Paul Wang	Georgetown University
Paulo Carvalho	Carnegie Mellon University
Pedro Manuel Moreno-Marcos	Universidad Carlos III de Madrid
Phillip Grimaldi	Khan Academy
Prateek Basavaraj	University of Central Florida
Rémi Venant	Le Mans Université - LIUM
Rene Kizilcec	Cornell University
Renza Campagni	Università degli Studi di Firenze
Roger Nkambou	Université du Québec À Montréal (UQAM)
Scott Crossley	Georgia State University
Sébastien Iksal	Université du Mans
Sébastien Lallé	The University of British Columbia
Sergey Sosnovsky	Utrecht University
Shahab Boumi	University of Central Florida
Shalini Pandey	University of Minnesota
Shitian Shen	North Carolina State University
Solmaz Abdi	The University of Queensland
Sotiris Kotsiantis	University of Patras
Spyridon Doukakis	Ionian University
Sreecharan Sankaranarayanan	National Institute of Technology Karnataka, Surathkal
Stefan Slater	University of Pennsylvania

Stephen Hutt	University of Pennsylvania
Tanja Käser	École Polytechnique Fédérale de Lausanne (EPFL)
Teresa Ober	University of Notre Dame
Thomas Price	North Carolina State University
Tounwendyam Frédéric Ouedraogo	Université Norbert Zongo
Tuyet-Trinh Vu	Hanoi University of Science and Technology
Vanessa Echeverria	Escuela Superior Politécnica del Litoral
Vasile Rus	The University of Memphis
Victor Menendez-Dominguez	Universidad Autónoma de Yucatán
Violetta Cavalli-Sforza	Al Akhawayn University, Morocco
Vladimir Ivančević	University of Novi Sad
Wenbin Zhang	Carnegie Mellon University
Yang Jiang	Educational Testing Service
Yang Shi	North Carolina State University
Yi-Jung Wu	University of Wisconsin - Madison
Yingbin Zhang	University of Illinois Urbana-Champaign
Yomna M.I. Hassan	Misr International University
Zhuqian Zhou	Teachers College, Columbia University

Sponsors

Silver



Bronze



Department of Computer Science



School of Education

Contributors



Best Paper Selection

The program committee chairs discussed and nominated four full papers and four short papers for best paper and best student paper awards, based on reviews, review scores, and meta-reviews. The papers and reviews (both anonymous) were then sent to a best paper award committee who ranked the papers. The highest-ranked paper was awarded the best paper award, while the next-highest ranked paper with a student first author was awarded the best student paper award.

Best paper committee

- Luc Paquette
- Kalina Yacef
- Kenneth Koedinger
- Anna Rafferty

Best paper nominees

(Full) Jiayi Zhang, Juliana Ma, Alexandra L. Andres, Stephen Hutt, Ryan S. Baker, Jaclyn Ocumpaugh, Caitlin Mills, Jamiella Brooks, Sheela Sethuraman and Tyron Young. *Detecting SMART Model Cognitive Operations in Mathematical Problem-Solving Process*

(Full) Vinthuy Phan, Laura Wright and Bridgette Decent. *Addressing Competing Objectives in Allocating Funds to Scholarships and Need-based Financial Aid*

(Full) Yuyang Nie, Helene Deacon, Alona Fyshe and Carrie Demmans Epp. *Predicting Reading Comprehension Scores of Elementary School Students*

(Full) Guojing Zhou, Robert Moulder, Chen Sun and Sidney K. D’Mello. *Investigating Temporal Dynamics Underlying Successful Collaborative Problem Solving Behaviors with Multilevel Vector Autoregression*

(Short) Lea Cohausz. *Towards Real Interpretability of Student Success Prediction Combining Methods of XAI and Social Science*

(Short) Juan Sanguino, Ruben Manrique, Olga Mariño, Mario Linares and Nicolas Cardozo. *Log mining for course recommendation in limited information scenarios*

(Short) Zhikai Gao, Bradley Erickson, Yiqiao Xu, Collin Lynch, Sarah Heckman and Tiffany Barnes. *Admitting you have a problem is the first step: Modeling when and why students seek help in programming assignments*

(Short) Anaïs Tack and Chris Piech. *The AI Teacher Test: Measuring the Pedagogical Ability of Blender and GPT-3 in Educational Dialogues*

Table of Contents

Frontmatter

Preface	iii
Organizing Committee	v
Sponsors	xii
Best Paper Selection	xiii

Abstracts

Keynotes	1
JEDM Presentations	3

Long Papers

Insta-Reviewer: A Data-Driven Approach for Generating Instant Feedback on Students’ Project Reports	5
<i>Qinjin Jia, Mitchell Young, Yunkai Xiao, Jialin Cui, Chengyuan Liu, Parvez Rashid and Edward Gehringer</i>	
Sparse Factor Autoencoders for Item Response Theory	17
<i>Benjamin Paaßen, Malwina Dywel, Melanie Fleckenstein and Niels Pinkwart</i>	
Exploring Common Trends in Online Educational Experiments	27
<i>Ethan Prihar, Manaal Syed, Korinn Ostrow, Stacy Shaw, Adam Sales and Neil Heffernan</i>	
Optimizing Representations and Policies for Question Sequencing using Reinforcement Learning	39
<i>Aqil Zainal Azhar, Avi Segal and Kobi Gal</i>	
Code-DKT: A Code-based Knowledge Tracing Model for Programming Tasks	50
<i>Yang Shi, Min Chi, Tiffany Barnes and Thomas Price</i>	
Building a Reinforcement Learning Environment from Limited Data to Optimize Teachable Robot Interventions	62
<i>Tristan Maidment, Mingzhi Yu, Nikki Lobczowski, Adriana Kovashka, Erin Walker, Diane Litman and Timothy Nokes-Malach</i>	
Detecting SMART Model Cognitive Operations in Mathematical Problem-Solving Process	75
<i>Jiayi Zhang, Juliana Ma, Alexandra L. Andres, Stephen Hutt, Ryan S. Baker, Jaclyn Ocumpaugh, Caitlin Mills, Jamiella Brooks, Sheela Sethuraman and Tyron Young</i>	
SQL-DP: A Novel Difficulty Prediction Framework for SQL Programming Problems	86
<i>Jia Xu, Tingting Wei and Pin Lv</i>	
Evaluating the Explainers: Black-Box Explainable Machine Learning for Student Success Prediction in MOOCs	98

<i>Vinitra Swamy, Bahar Radmehr, Natasa Krco, Mirko Marras and Tanja Käser</i>	
Addressing Competing Objectives in Allocating Funds to Scholarships and Need-based Financial Aid	110
<i>Vinthuy Phan, Laura Wright and Bridgette Decent</i>	
Automatic Short Math Answer Grading via In-context Meta-learning	122
<i>Mengxue Zhang, Sami Baral, Neil Heffernan and Andrew Lan</i>	
Investigating Multimodal Predictors of Peer Satisfaction for Collaborative Coding in Middle School	133
<i>Yingbo Ma, Gloria Ashiya Katuka, Mehmet Celepkolu and Kristy Elizabeth Boyer</i>	
Going Deep and Far: Gaze-based Models Predict Multiple Depths of Comprehension During and One Week Following Reading	145
<i>Megan Caruso, Candace Peacock, Rosy Southwell, Guojing Zhou and Sidney D’Mello</i>	
Predicting Reading Comprehension Scores of Elementary School Students	158
<i>Yuyang Nie, Helene Deacon, Alona Fyshe and Carrie Demmans Epp</i>	
Enhancing Stealth Assessment in Game-Based Learning Environments with Generative Zero-Shot Learning	171
<i>Nathan Henderson, Halim Acosta, Wookhee Min, Bradford Mott, Trudi Lord, Frieda Reichsman, Chad Dorsey, Eric Wiebe and James Lester</i>	
Generalisable Methods for Early Prediction in Interactive Simulations for Education	183
<i>Jade Mai Cock, Mirko Marras, Christian Giang and Tanja Käser</i>	
Toward Better Grade Prediction via A2GP - An Academic Achievement Inspired Predictive Model	195
<i>Wei Qiu, S. Supraja and Andy W. H. Khong</i>	
Individual Fairness Evaluation for Automated Essay Scoring System	206
<i>Afrizal Doewes, Akрати Saxena, Yulong Pei and Mykola Pechenzkiy</i>	
Combining domain modelling and student modelling techniques in a single automated pipeline	217
<i>Gio Picones, Benjamin Paaßen, Irena Koprinska and Kalina Yacef</i>	
Neural Recall Network: A Neural Network Solution to Low Recall Problem in Regex-based Qualitative Coding	228
<i>Zhiqiang Cai, Cody Marquart and David Shaffer</i>	
Towards Including Instructor Features in Student Grade Prediction	239
<i>Nathan Ong, Jiaye Zhu and Daniel Mosse</i>	
Item Response Theory-Based Gaming Detection	251
<i>Yun Huang, Steven Dang, J. Elizabeth Richey, Michael Asher, Nikki G. Lobczowski, Danielle Chine, Elizabeth A. McLaughlin, Judith M. Harackiewicz, Vincent Alevin and Kenneth Koedinger</i>	
Exploring Cultural Diversity and Collaborative Team Communication through a Dynamical Systems Lens	263

Mohammad Amin Samadi, Jacqueline G. Cavazos, Yiwen Lin and Nia Nixon

[Predicting Cognitive Engagement in Online Course Discussion Forums](#) 276
Guher Gorgun, Seyma Nur Yildirim-Erbasli and Carrie Demmans Epp

[Investigating Temporal Dynamics Underlying Successful Collaborative Problem Solving Behaviors with Multilevel Vector Autoregression](#) 290
Guojing Zhou, Robert Moulder, Chen Sun and Sidney D’Mello

[Challenges and Feasibility of Automatic Speech Recognition for Modeling Student Collaborative Discourse in Classrooms](#) 302
Rosy Southwell, Samuel Pugh, E. Margaret Perkoff, Charis Clevenger, Jeffrey Bush, Rachel Lieber, Wayne Ward, Peter Foltz and Sidney D’Mello

Short Papers

[Does Practice Make Perfect? Analyzing the Relationship Between Higher Mastery and Forgetting in an Adaptive Learning System](#) 316
Jeffrey Matayoshi, Eric Cosyn and Hasan Uzun

[Improving Peer Assessment with Graph Neural Networks](#) 325
Alireza A. Namanloo, Julie Thorpe and Amirali Salehi-Abari

[Using Neural Network-Based Knowledge Tracing for a Learning System with Unreliable Skill Tags](#) 333
Shamya Karumbaiah, Jiayi Zhang, Ryan Baker, Richard Scruggs, Whitney Cade, Margaret Clements and Shuqiong Lin

[#lets-discuss: Analyzing Student Affect in Course Forums Using Emoji](#) 339
Ariel Blobstein, Kobi Gal, David Karger, Marc Facciotti, Hyunsoo Kim, Jumana Almahmoud and Kamali Sripathi

[Investigating Growth of Representational Competencies by Knowledge-Component Model](#) . 346
Jihyun Rho, Martina Rau and Barry Vanveen

[Adversarial bandits for drawing generalizable conclusions in non-adversarial experiments: an empirical study](#) 353
Yang Zhi-Han, Shiyue Zhang and Anna Rafferty

[Towards Real Interpretability of Student Success Prediction Combining Methods of XAI and Social Science](#) 361
Lea Cohausz

[An Evaluation of code2vec Embeddings for Scratch](#) 368
Benedikt Fein, Isabella Graßl, Florian Beck and Gordon Fraser

[Investigating the effect of Automated Feedback on learning behavior in MOOCs for programming](#) 376
Hagit Gabbay and Anat Cohen

[Data-driven goal setting: Searching optimal badges in the decision forest](#) 384

Julian Langenhagen

Improving problem detection in peer assessment through pseudo-labeling using semi-supervised learning	391
<i>Chengyuan Liu, Jialin Cui, Ruixuan Shang, Yunkai Xiao, Qinjin Jia and Edward Gehringer</i>	
Evaluating Gaming Detector Model Robustness Over Time	398
<i>Nathan Levin, Ryan Baker, Nidhi Nasiar, Stephen Fancsali and Stephen Hutt</i>	
Characterizing joint attention dynamics during collaborative problem-solving in an immersive astronomy simulation	406
<i>Yiqiu Zhou and Jina Kang</i>	
Can Population-based Engagement Improve Personalisation? A Novel Dataset and Experiments	414
<i>Sahan Bulathwela, Meghana Verma, María Pérez Ortiz, Emine Yilmaz and John Shawe-Taylor</i>	
Is there Method in Your Mistakes? Capturing Error Contexts by Graph Mining for Targeted Feedback	422
<i>Maximilian Jahnke and Frank Höppner</i>	
Log mining for course recommendation in limited information scenarios	430
<i>Juan Sanguino, Ruben Manrique, Olga Mariño, Mario Linares and Nicolas Cardozo</i>	
Using Machine Learning Explainability Methods to Personalize Interventions for Students	438
<i>Paul Hur, HaeJin Lee, Suma Bhat and Nigel Bosch</i>	
Grade Prediction via Prior Grades and Text Mining on Course Descriptions: Course Outlines and Intended Learning Outcomes	446
<i>Jiawei Li, S. Supraja, Wei Qiu and Andy W. H. Khong</i>	
From {Solution} Synthesis to {Student Attempt} Synthesis for Block-Based Visual Programming Tasks	454
<i>Adish Singla and Nikitas Theodoropoulos</i>	
Mining Assignment Submission Time to Detect At-Risk Students with Peer Information	462
<i>Yuancheng Wang, Nanyu Luo and Jianjun Zhou</i>	
Simulating Policy Changes in Prerequisite-Free Curricula: A Supervised Data-Driven Approach	470
<i>Frederik Baucks and Laurenz Wiskott</i>	
Modeling One-on-one Online Tutoring Discourse using an Accountable Talk Framework	477
<i>Renu Balyan, Tracy Arner, Karen Taylor, Jinnie Shin, Michelle Banawan, Walter Leite and Danielle McNamara</i>	
Using Markov Models and Random Walks to Examine Strategy Use of More or Less Successful Comprehenders	484
<i>Katerina Christhilf, Natalie Newton, Reese Butterfuss, Kathryn S. McCarthy, Laura K. Allen, Joseph P. Magliano and Danielle S. McNamara</i>	

No Meaning Left Unlearned: Predicting Learners’ Knowledge of Atypical Meanings of Words from Vocabulary Tests for Their Typical Meanings	492
<i>Yo Ehara</i>	
Using community-based problems to increase motivation in a data science virtual internship	500
<i>Jillian Johnson and Andrew Olney</i>	
Admitting you have a problem is the first step: Modeling when and why students seek help in programming assignments.	508
<i>Zhikai Gao, Bradley Erickson, Yiqiao Xu, Collin Lynch, Sarah Heckman and Tiffany Barnes</i>	
Going beyond “Good Job”: Analyzing Helpful Feedback from the Student’s Perspective	515
<i>M Parvez Rashid, Yunkai Xiao and Edward F. Gehringer</i>	
The AI Teacher Test: Measuring the Pedagogical Ability of Blender and GPT-3 in Educational Dialogues	522
<i>Anaïs Tack and Chris Piech</i>	
Automatic Classification of Learning Objectives Based on Bloom’s Taxonomy	530
<i>Yuheng Li, Mladen Rakovic, Boon Xin Poh, Dragan Gasevic and Guanliang Chen</i>	
Posters	
Skills Taught vs Skills Sought: Using Skills Analytics to Identify the Gaps between Curriculum and Job Markets	538
<i>Alireza Ahadi, Kirsty Kitto, Marian-Andrei Rizoiu and Katarzyna Musial</i>	
DeepIRT with a Hypernetwork to Optimize the Degree of Forgetting of Past Data	543
<i>Emiko Tsutsumi, Yiming Guo and Maomi Ueno</i>	
Mining and Assessing Anomalies in Students’ Online Learning Activities with Self-supervised Machine Learning	549
<i>Lan Jiang and Nigel Bosch</i>	
Faster Confidence Intervals for Item Response Theory via an Approximate Likelihood Profile	555
<i>Benjamin Paaßen, Christina Göpfert and Niels Pinkwart</i>	
Towards the understanding of cultural differences in between gamification preferences: A data-driven comparison between the US and Brazil	560
<i>Armando Toda, Ana Klock, Filipe Dwan Pereira, Luiz Antonio Rodrigues, Paula Toledo Palomino, Vinicius Lopes, Craig Stewart, Elaine H. T. Oliveira, Isabela Gasparini, Seiji Isotani and Alexandra Cristea</i>	
Supervised Machine Learning for Modelling STEM Career and Education Interest in Irish School Children	565
<i>Annika Lindh, Keith Quille, Aidan Mooney, Kevin Marshall and Katriona O’Sullivan</i>	
Equitable Ability Estimation in Neurodivergent Student Populations with Zero-Inflated Learner Models	571
<i>Niall Twomey, Sarah McMullan, Anat Elhalal, Rafael Poyiadzi and Luis Vaquero</i>	

Equity and Fairness of Bayesian Knowledge Tracing	578
<i>Sebastian Tschiatschek, Maria Knobelsdorf and Adish Singla</i>	
Analyzing the Equity of the Brazilian National High School Exam by Validating the Item Response Theory’s Invariance	583
<i>Vitoria Guardieiro, Marcos M. Raimundo and Jorge Poco</i>	
A deep dive into microphone hardware for recording collaborative group work	588
<i>Mariah Bradford, Paige Hansen, J. Ross Beveridge, Nikhil Krishnaswamy and Nathaniel Blanchard</i>	
Linguistic Profiles of Students Interacting with Conversation-Based Assessment Systems . .	594
<i>Carol Forsyth, Jesse Sparks, Jonathan Steinberg and Laura McCulla</i>	
Does chronology matter? Sequential vs contextual approaches to knowledge tracing	601
<i>Yueqi Wang and Zachary Pardos</i>	
Algorithmic unfairness mitigation in student models: When fairer methods lead to unintended results	606
<i>Frank Stinar and Nigel Bosch</i>	
The Impact of Semester Gaps on Student Grades	612
<i>Gary Weiss, Joseph Denham and Daniel Leeds</i>	
Assessing Instructor Effectiveness Based on Future Student Performance	616
<i>Gary Weiss, Erik Brown, Michael Riad-Zaky, Ruby Iannone and Daniel Leeds</i>	
Modeling study duration considering course enrollments and student diversity	621
<i>Niels Seidel</i>	
Generalized Sequential Pattern Mining of Undergraduate Courses	629
<i>Daniel Leeds, Cody Chen, Yijun Zhao, Fiza Metla, James Guest and Gary Weiss</i>	
Employing Tree-based Algorithms to Predict Students’ Self-Efficacy in PISA 2018	634
<i>Bin Tan and Maria Cutumisu</i>	
Identifying Longitudinal Attendance Patterns through Student Subpopulation Distribution Comparison	640
<i>Zitong Zhao, Pan Deng and Jianjun Zhou</i>	
Improved Automated Essay Scoring using Gaussian Multi-Class SMOTE for Dataset Sampling	647
<i>Jih Soong Tan, Ian K. T. Tan, Lay Ki Soon and Huey Fang Ong</i>	
A Topic-Centric Crowdsourced Assisted Biomedical Literature Review Framework for Academics	652
<i>Ryan Hodgson, Jingyun Wang, Alexandra Cristea, Fumiko Matsuzaki and Hiroyuki Kubota</i>	
Personalized and Explainable Course Recommendations for Students at Risk of Dropping out	657
<i>Kerstin Wagner, Agathe Merceron, Petra Sauer and Niels Pinkwart</i>	
A deep reinforcement learning approach to automatic formative feedback	662
<i>Aubrey Condor and Zachary Pardos</i>	

Preliminary Experiments with Transformer based Approaches To Automatically Inferring Domain Models from Textbooks	667
<i>Rabin Banjade, Priti Oli, Lasang Jimba Tamang and Vasile Rus</i>	
E-learning Preparedness: A Key Consideration to Promote Fair Learning Analytics Development in Higher Education	673
<i>Jinnie Shin, Okan Bulut and Wallace N. Pinto Jr.</i>	
Leveraging Auxiliary Data from Similar Problems to Improve Automatic Open Response Scoring	679
<i>Raysa Rivera-Bergollo, Sami Baral, Anthony Botelho and Neil Heffernan</i>	
Modifying Deep Knowledge Tracing for Multi-step Problems	684
<i>Qiao Zhang, Zeyu Chen, Natasha Lalwani and Christopher MacLellan</i>	
Clustering Students Using Pre-Midterm Behaviour Data and Predict Their Exam Performance	689
<i>Huanyi Chen and Paul Ward</i>	
Format-Aware Item Response Theory for Predicting Vocabulary Proficiency	695
<i>Boxuan Ma, Gayan Prasad Hettiarachchi and Yuji Ando</i>	
Towards Automated Generation and Evaluation of Questions in Educational Domains	701
<i>Shravya Bhat, Huy Nguyen, Steven Moore, John Stamper, Majd Sakr and Eric Nyberg</i>	
MOOC-Rec: Instructional Video Clip Recommendation for MOOC Forum Questions	705
<i>Peide Zhu, Claudia Hauff and Jie Yang</i>	
Automatic Graph-based Knowledge Tracing	710
<i>Ting Long, Yunfei Liu, Weinan Zhang, Wei Xia, Zhicheng He, Ruiming Tang and Yong Yu</i>	
Process-BERT: A Framework for Representation Learning on Educational Process Data . . .	715
<i>Alexander Scarlatos, Christopher Brinton and Andrew Lan</i>	
Online Item Response Theory (OIRT) - Tracking Student Abilities in Online Learning System	720
<i>Luyao Peng and Chengzhi Wei</i>	
Looking for the best data fusion model in Smart Learning Environments for detecting at risk university students	725
<i>Cristobal Romero, Wilson Chango and Rebeca Cerezo</i>	
Distance measure between instructor-recommended and learner's learning pathways	729
<i>Marie-Luce Bourguet and Yushan Li</i>	
Student Perception on the Effectiveness of On-Demand Assistance in Online Learning Platforms	734
<i>Aaron Haim and Neil Heffernan</i>	
Theory-Informed Problem-Solving Sequential Pattern Visualization	738
<i>Zilong Pan and Min Liu</i>	
CHUNK Learning: A Tool that Supports Personalized Education	743
<i>Raluca Gera, D'Marie Bartolf, Simona Tick and Akрати Saxena</i>	

Do College Students Learn Math the Same Way as Middle School Students? On the Transferability of Findings on Within-Problem Supports in Intelligent Tutoring Systems	748
<i>Michael Smalenberger and Kelly Smalenberger</i>	
Comparison of Learning Behaviors on an e-Book System in 2019 Onsite and 2020 Online Courses	753
<i>Hiroaki Kawashima</i>	
Recommendation System of Mobile Language Learning Applications: Similarity versus Diversity in Learner Preference	758
<i>Juyeong Song, Kisu Yang, Hyeji Jang and Hyo-Jeong So</i>	
A Variant of Performance Factors Analysis Model for Categorization	763
<i>Meng Cao and Philip Pavlik</i>	
Selecting Reading Texts Suitable for Incidental Vocabulary Learning by Considering the Estimated Distribution of Acquired Vocabulary	767
<i>Yo Ehara</i>	
Doctoral Consortium	
Identifying Explanations Within Student-Tutor Chat Logs	773
<i>Ethan Prihar, Alexander Moore and Neil Heffernan</i>	
Detecting When a Learner Requires Assistance with Programming and Delivering a Useful Hint	778
<i>Marcus Messer</i>	
A Paraphrase Identification Approach in Paragraph length texts	782
<i>Arwa Al Saqaabi, Craig Stewart, Eleni Akrida and Alexandra Cristea</i>	
Investigating learners' Cognitive Engagement in Python Programming using ICAP framework	789
<i>Daevesh Singh and Ramkumar Rajendran</i>	
Improving Automated Assessment and Feedback for Student Open-responses in Mathematics	795
<i>Sami Baral</i>	
Modeling Cognitive Load and Affect to Support Adaptive Online Learning	799
<i>Minghao Cai and Carrie Demmans Epp</i>	
Using AI, ML and Sentiment Analysis to Increase Diversity and Equity in Technology Training and Careers	805
<i>Jonathan Young, Sue Black, Alexandra Cristea, Ryan Hodgson and Cristina Todor</i>	
Effect of Q-matrix Misspecification on Variational Autoencoders (VAE) for Multidimensional Item Response Theory (MIRT) Models Estimation	811
<i>Mahbul Hasan, Lih Y Deng, John Sabatini, Dale Bowman, Ching-Chi Yang and John Hollander</i>	
Towards Personalised Learning of Psychomotor Skills with Data Mining	816
<i>Miguel Portaz and Olga C. Santos</i>	

Industry Track

Using a Randomized Experiment to Compare the Performance of Two Adaptive Assessment Engines	821
<i>Jeffrey Matayoshi, Hasan Uzun and Eric Cosyn</i>	
Mining Artificially Generated Data to Estimate Competency	828
<i>Robby Robson, Benjamin Goldberg, Shelly Blake-Plock, Cliff Casey, William Hoyt, Mike Hernandez and Fritz Ray</i>	
“Closing the Loop” in Educational Data Science with an Open Source Architecture for Large-Scale Field Trials	834
<i>Stephen Fancsali, April Murphy and Steven Ritter</i>	
Estimating the causal effects of Khan Academy Map Accelerator across demographic sub-groups	839
<i>Phillip Grimaldi, Kodi Weatherholtz and Kelli Millwood Hill</i>	

Workshops & Tutorials

The Third Workshop of The Learner Data Institute: Big Data, Research Challenges, & Science Convergence in Educational Data Science	847
<i>Vasile Rus and Stephen Fancsali</i>	
FATED 2022: Fairness, Accountability, and Transparency in Educational Data	848
<i>Collin Lynch, Mirko Marras, Mykola Pechenizkiy, Anna Rafferty, Steve Ritter, Vinitra Swamy and Renzhe Yu</i>	
6th Educational Data Mining in Computer Science Education (CSEDM) Workshop	850
<i>Bitu Akram, Thomas Price, Yang Shi, Peter Brusilovsky and Sharon I-Han</i>	
Rethinking Accessibility: Applications in Educational Data Mining	851
<i>Juanita Hicks, Ruhan Circi, Burhan Ogut, Michelle Yin and Darrick Yee</i>	
Causal Inference in Educational Data Mining	852
<i>Adam Sales and Neil Heffernan</i>	
Using the Open Science Framework to promote Open Science in Education Research	853
<i>Stacy Shaw and Adam Sales</i>	

Keynotes

Deep Down, Everyone Wants to be Causal

Jennifer Hill, Professor of Applied Statistics at New York University, USA

Most researchers in the social, behavioral, and health sciences are taught to be extremely cautious in making causal claims. However, causal inference is a necessary goal in research for addressing many of the most pressing questions around policy and practice. In the past decade, causal methodologists have increasingly been using and touting the benefits of more complicated machine learning algorithms to estimate causal effects. These methods can take some of the guesswork out of analyses, decrease the opportunity for “p-hacking,” and may be better suited for more fine-tuned tasks such as identifying varying treatment effects and generalizing results from one population to another. However, should these more advanced methods change our fundamental views about how difficult it is to infer causality? In this talk, I will discuss some potential advantages and disadvantages of using machine learning for causal inference and emphasize ways that we can all be more transparent in our inferences and honest about their limitations.

Beyond Algorithmic Fairness in Education: Equitable and Inclusive Decision-Support Systems

René Kizilcec, Assistant Professor of Information Science at Cornell University, USA

Advancing equity and inclusion in schools and universities has long been a priority in education research. While data-driven predictive models could help improve social injustices in education, many studies from other domains suggest instead that these models tend to exacerbate existing inequities without added precautions. A growing body of research from the educational data mining and neighboring communities is beginning to map out where biases are likely to occur, what contributes to them, and how to mitigate them. These efforts to advance algorithmic fairness are an important research direction, but it is critical to also consider how AI systems are used in educational contexts to support decisions and judgements. In this talk, I will survey research on algorithmic fairness and explore the role of human factors in AI systems and their implications for advancing equity and inclusion in education.

No data about me without me: Including Learners and Teachers in Educational Data Mining

Judy Robertson, Professor of Digital Learning at the University of Edinburgh, Scotland

The conference theme this year emphasizes the broadening of participation and inclusion in educational data mining; in this talk, I will discuss methodologies for including learners and teachers throughout the research process. This involves not only preventing harm to young learners which might result from insufficient care when processing their data but also embracing their participation in the design and evaluation of educational data mining technologies. I will argue that even young learners can and should be included in the analysis and interpretation of data which affects them. I will give examples of a project in which children have the role of data activists, using

classroom sensor data to explore their readiness to learn.

Test of Time Award:

Compassionate, Data-Driven Tutors for Problem Solving and Persistence

Tiffany Barnes, Distinguished Professor of Computer Science at North Carolina State University, USA

Determining how, when, and whether to provide personalized support is a well-known challenge called the assistance dilemma. A core problem in solving the assistance dilemma is the need to discover when students are unproductive so that the tutor can intervene. This is particularly challenging for open-ended domains, even those that are well-structured with defined principles and goals. In this talk, I will present a set of data-driven methods to classify, predict, and prevent unproductive problem-solving steps in the well-structured open-ended domains of logic and programming. Our approaches leverage and extend my work on the Hint Factory, a set of methods that to build data-driven intelligent tutor supports using prior student solution attempts. In logic, we devised a HelpNeed classification model that uses prior student data to determine when students are likely to be unproductive and need help learning optimal problem-solving strategies. In a controlled study, we found that students receiving proactive assistance on logic when we predicted HelpNeed were less likely to avoid hints during training, and produced significantly shorter, more optimal posttest solutions in less time. In a similar vein, we have devised a new data-driven method that uses student trace logs to identify struggling moments during a programming assignment and determine the appropriate time for an intervention. We validated our algorithm's classification of struggling and progressing moments with experts rating whether they believe an intervention is needed for a sample of 20% of the dataset. The result shows that our automatic struggle detection method can accurately detect struggling students with less than 2 minutes of work with 77% accuracy. We further evaluated a sample of 86 struggling moments, finding 6 reasons that human tutors gave for intervention from missing key components to needing confirmation and next steps. This research provides insight into the when and why for programming interventions. Finally, we explore the potential of what supports data-driven tutors can provide, from progress tracking to worked examples and encouraging messages, and their importance for compassionately promoting persistence in problem solving.

JEDM Presentations

Empirical Evaluation of Deep Learning Models for Knowledge Tracing: Of Hyperparameters and Metrics on Performance and Replicability

Sami Sarsa Aalto University, Finland

Juho Leinonen Aalto University, Finland

Arto Hellas Aalto University, Finland

New knowledge tracing models are continuously being proposed, even at a pace where state-of-the-art models cannot be compared with each other at the time of publication. This leads to a situation where ranking models is hard, and the underlying reasons of models' performance – be it architectural choices, hyperparameter tuning, performance metrics, or data – is often underexplored. In this work, we review and evaluate a body of deep learning knowledge tracing (DLKT) models with openly available and widely-used data sets, and with a novel data set of students learning to program. The evaluated knowledge tracing models include Vanilla-DKT, two Long Short-Term Memory Deep Knowledge Tracing (LSTM-DKT) variants, two Dynamic Key-Value Memory Network (DKVMN) variants, and Self-Attentive Knowledge Tracing (SAKT). As baselines, we evaluate simple non-learning models, logistic regression and Bayesian Knowledge Tracing (BKT). To evaluate how different aspects of DLKT models influence model performance, we test input and output layer variations found in the compared models that are independent of the main architectures. We study maximum attempt count options, including filtering out long attempt sequences, that have been implicitly and explicitly used in prior studies. We contrast the observed performance variations against variations from non-model properties such as randomness and hardware. Performance of models is assessed using multiple metrics, whereby we also contrast the impact of the choice of metric on model performance. The key contributions of this work are the following: Evidence that DLKT models generally outperform more traditional models, but not necessarily by much and not always; Evidence that even simple baselines with little to no predictive value may outperform DLKT models, especially in terms of accuracy – highlighting importance of selecting proper baselines for comparison; Disambiguation of properties that lead to better performance in DLKT models including metric choice, input and output layer variations, common hyperparameters, random seeding and hardware; Discussion of issues in replicability when evaluating DLKT models, including discrepancies in prior reported results and methodology. Model implementations, evaluation code, and data are published as a part of this work.

Latent Skill Mining and Labeling from Courseware Content

Noboru Matsuda North Carolina State University, USA
Jesse Wood North Carolina State University, USA
Raj Shrivastava North Carolina State University, USA
Machi Shimmei North Carolina State University, USA
Norman Bier Carnegie Mellon University, USA

A model that maps the requisite skills, or knowledge components, to the contents of an online course is necessary to implement many adaptive learning technologies. However, developing a skill model and tagging courseware contents with individual skills can be expensive and error prone. We propose a technology to automatically identify latent skills from instructional text on existing online courseware called SMART (Skill Model mining with Automated detection of Resemblance among Texts). SMART is capable of mining, labeling, and mapping skills without using an existing skill model or student learning (aka response) data. The goal of our proposed approach is to mine latent skills from assessment items included in existing courseware, provide discovered skills with human-friendly labels, and map didactic paragraph texts with skills. This way, mapping between assessment items and paragraph texts is formed. In doing so, automated skill models produced by SMART will reduce the workload of courseware developers while enabling adaptive online content at the launch of the course. In our evaluation study, we applied SMART to two existing authentic online courses. We then compared machine-generated skill models and human-crafted skill models in terms of the accuracy of predicting students' learning. We also evaluated the similarity between machine-generated and human-crafted skill models. The results show that student models based on SMART-generated skill models were equally predictive of students' learning as those based on human-crafted skill models – as validated on two OLI courses. Also, SMART can generate skill models that are highly similar to human-crafted models as evidenced by the normalized mutual information (NMI) values.

Insta-Reviewer: A Data-Driven Approach for Generating Instant Feedback on Students' Project Reports

Qinjin Jia, Mitchell Young, Yunkai Xiao, Jialin Cui, Chengyuan Liu, Parvez Rashid, Edward Gehringer
Department of Computer Science
North Carolina State University
Raleigh, NC, USA
qjia3, mgyoung, yxiao28, jcui9, cliu32, mrashid4, efg@ncsu.edu

ABSTRACT

Providing timely feedback is crucial in promoting academic achievement and student success. However, for multifarious reasons (e.g., limited teaching resources), feedback often arrives too late for learners to act on the feedback and improve learning. Thus, automated feedback systems have emerged to tackle educational tasks in various domains, including novice programming, short-essay writing, and open-ended questions. However, to the best of our knowledge, no previous study has investigated automated feedback generation on students' project reports. In this paper, we present a novel data-driven system, named *Insta-Reviewer*, for automatically generating instant feedback on students' project reports, using state-of-the-art natural language processing (NLP) models. We also propose a framework for manually evaluating system-generated feedback. Experimental results show that feedback generated by Insta-Reviewer on real students' project reports can achieve near-human performance. Our work demonstrates the feasibility of automatic feedback generation for students' project reports while highlighting several prominent challenges for future research.

Keywords

Feedback generation, automated review generation, instant feedback, learning at scale, mining educational data

1. INTRODUCTION

Feedback plays a vital role in the student learning process, as it can help students reinforce or correct their understanding of knowledge and content by giving them clear guidance on how to improve their learning [1, 19, 22, 39]. Furthermore, instant feedback is usually more effective than delayed feedback, presumably because timely feedback is more likely to motivate students to stay on task and encourage them to achieve learning goals [21, 51, 14]. However, owing to various constraints (e.g., staff availability), feedback often comes too late for students to enact the advice and benefit their learning [48, 38, 32, 22]. Students reported in a prior study

Q. Jia, M. Young, Y. Xiao, J. Cui, C. Liu, P. Rashid, and E. Gehringer. Insta-reviewer: A data-driven approach for generating instant feedback on students' project reports. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 5–16, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853099>

that delayed feedback is perceived as irrelevant because it has been so long that they have forgotten about the content, which discourages them from following the feedback [38]. Thus, tardy feedback can unintentionally position students as passive recipients of feedback information and limit their engagement with feedback and learning [8, 38].

One way of bringing about the much-needed immediacy is by way of automatic generation of instant feedback on students' work. Thanks to recent technological advancements, a variety of automatic feedback systems have emerged to tackle educational tasks in various domains, including novice programming [31, 56], short-essay writing [49, 5], and open-ended short answers [28, 3]. For instance, Malik and Wu proposed generative models for providing feedback on short answers and different types of programming assignments [28]. Marwan et al. designed a hybrid method to deliver instant feedback for block-based programming [31]. In addition to this, many other impressive studies focus on educational tasks that demand instant feedback to facilitate students' learning and show promising results across modalities and domains (e.g., [27, 36, 47, 2, 29]). It can be argued that automatic feedback systems will be integral parts of the future AI-powered educational ecosystem [41].

However, to the best of our knowledge, no attempt has been made to evaluate the feasibility of automatic feedback generation on students' project reports. It is well-known that course projects are an essential part of many university curricula, especially STEM courses [4, 17]. These projects can help students reinforce their theoretical knowledge and develop a host of skills that are increasingly important in the professional world [30, 17]. However, delivering immediate feedback on project reports is often infeasible for instructors. We summarize the reasons why such an automated feedback system for students' project reports is significant as follows:

1. Despite the fact that course projects have many positive educational impacts on students, the burden of providing timely feedback may prevent instructors from offering sufficient course projects. In this case, an automated feedback system can encourage instructors to provide more project work in classes.
2. Many instructors can merely provide summative feedback for a final project at the end of a semester, which does not give students an opportunity to implement the advice. However, if an automated feedback system is available to provide formative feedback, students will

have guidance on how to revise their work and reinforce their learning without adding workload to instructors.

3. An automated feedback system can help promote educational equity and diversity by giving students the benefit of quality feedback on projects in an institution that has high teaching loads and limited or nonexistent teaching assistant support, or even in a MOOC.

In this paper, we present a data-driven system, named *Insta-Reviewer*, for generating instant textual feedback on students’ project reports. Insta-Reviewer utilizes a select-then-generate paradigm consisting of two main steps: 1) the paradigm first uses an unsupervised method, called cross-entropy extraction, to summarize original reports to lengths acceptable for input into our text generation model used in the second step, and then 2) employs a supervised text-to-text generation model called BART to generate plausible and readable textual feedback for the corresponding report. In order to explore the quality of generated feedback, we employ a comprehensive set of evaluation metrics, including a content-overlap metric ROUGE, a model-based metric BERTScore, and a new human-centered evaluation metric.

To investigate the potential promise of our system, we design experiments to answer the following research questions:

RQ1: How effective is the proposed approach for generating feedback on students’ project reports?

RQ2: What are the problems of system-generated reviews? What are they not good at?

RQ3: How does the system perform compared with other potential methods?

RQ4: How does the automated feedback system perform in different small-data settings?

RQ5: Does the Insta-Reviewer automated feedback system raise any ethical concerns?

Our results show that feedback generated by Insta-Reviewer on real students’ project reports can achieve near-human performance, while it may include some non-factual or ambiguous statements in generated feedback. Our work demonstrates the feasibility of automated instant-feedback generation on students’ project reports. Experimental results also highlight several major challenges for future research.

Our main contributions are: 1) we present an effective data-driven approach for generating feedback on students’ project reports; 2) we collect a new dataset of students’ reports and expert reviews to facilitate future research endeavors; 3) we propose a new framework for manually evaluating generated feedback; 4) we evaluate the effectiveness of our approach in different small-data settings to help others who intend to apply the approach to their datasets; 5) we highlight several prominent challenges for future research.

The remainder of the paper is organized as follows: Section 2 presents related work. Section 3 describes the dataset used for this study. Section 4 elaborates our methodology for automatically generating feedback for students’ project reports and explains the new human-evaluation metric. Section 5 presents and discusses our experimental results. Section 6 concludes the paper, mentions the limitations of our work and provides some discussion about future research.

2. LITERATURE REVIEW

In the following, we first review prior studies on automated feedback systems. Then we survey potential metrics for evaluating system-generated feedback. Finally, we review previous work related to ethical concerns in feedback generation.

2.1 Automated Feedback Systems

In the field of education, feedback is defined as information provided by an agent (e.g., teacher, peer) about a learner’s performance or understanding, and it is one of the most significant influences on student learning and achievement [19]. Previous research has been devoted to designing a variety of automated feedback systems that provide feedback on various forms of student work, such as short-answer questions, essays, and programming problems. Although these efforts were not intended to provide feedback on student project reports, we reviewed these studies to gain some insight. The feedback-generation models (i.e., the feedback engines) used in previous studies on automated feedback systems for student work can be broadly categorized into two groups:

Expert-driven methods: Expert-driven methods (also called rule-based methods) use a set of expert-designed rules to provide feedback. For instance, Narciss et al. [34] presented an intelligent tutoring system for students learning mathematics. The system uses a set of pre-defined rules provided by domain experts to deliver feedback for students’ answers to numerical or multiple-choice questions. Nagata et al. [33] introduced an approach for leveraging the expert-driven method to diagnose preposition errors and produce feedback for learners’ English writing. While expert-driven methods are typically accurate and not data-hungry, they are not suitable for dealing with complex open-ended problems (e.g., generating feedback for students’ project reports) because creating and maintaining a vast set of expert-design rules for such open-ended problems is nearly impossible. Additionally, these methods usually produce feedback that is limited to fixed expressions without dynamic explanations.

Data-driven methods: Recent technological advances in artificial intelligence have enabled the development of various data-driven automated feedback systems to produce feedback for more complex open-ended tasks. Data-driven methods generate feedback by learning the mappings (i.e., patterns) from student work to expert feedback by means of machine-learning or deep-learning algorithms [10]. For example, Lu and Cutumisu [27] implemented several deep-learning models, including CNN, CNN+LSTM, and CNN+Bi-LSTM, for generating textual feedback on students’ essays. However, traditional deep-learning models usually fail to capture long-span dependencies in long documents and rely on large amounts of training data. More recent work has begun to use large-scale pre-trained language models, such as BERT [11], BART [23], and GPT-2 [40], for generating feedback on open-ended student work. These language models use the attention mechanism [45] to learn long-span dependencies and are pre-trained on large generic corpora in an unsupervised manner to reduce the need for labeled data. For instance, Olney [35] attempted to generate elaborated feedback for student responses using the ELI5 model and achieved promising results. In this paper, we also use such a pre-trained language model to exploit its ability to capture long-span dependencies in student reports.

2.2 Evaluation of Feedback Generation

Effective metrics for evaluating generated feedback are essential since we use them to compare different approaches and quantify the progress made on this research problem. However, unlike other tasks (e.g., text classification), accurately evaluating system-generated feedback (and many other natural language generation problems) is in itself a huge challenge, mainly because generating feedback is a highly open-ended task. For instance, an automated feedback system can generate multiple plausible reviews for the same student report, but all these reviews can be vastly different.

All existing evaluation methods for natural language generation tasks can be grouped into three categories: 1) content-overlap metrics, 2) model-based metrics, and 3) human-centered evaluation metrics. Content-overlap metrics and model-based metrics automatically evaluate a text-generation system by measuring the similarity between generated texts and reference texts provided by domain experts. Human-centered evaluation asks people to assess the quality of system-generated texts against set task-specific criteria [9].

It is worth noting that the ultimate goal of our Insta-Reviewer automated feedback system is to generate feedback that is valuable to students instead of generating the exact same feedback as provided by instructors. For this reason, human-authored evaluation should be viewed as the gold standard when evaluating generated feedback. However, human evaluations are inconsistent and subjective, which can lead to erroneous conclusions or prevent researchers from comparing results across systems [9]. Thus, we also employ a content-overlap metric and a model-based metric to validate our human-evaluation results. In the following paragraphs, we survey potential metrics that can be applied to our task.

Content-overlap metrics: Content-overlap metrics refer to a set of metrics that evaluate the quality of generation by comparing generated texts with reference texts (e.g., ground-truth feedback provided by instructors) based on the content overlap, such as n -gram match. Despite the fact that this set of metrics has many limitations (e.g., they do not take synonyms into account), text-generation research usually uses metrics from this set for benchmark analysis with models since they are objective and fast to calculate. Two of the most commonly used content-overlap metrics for evaluating generated text are BLEU [37] and ROUGE [25].

BLEU (the Bilingual Evaluation Understudy) is a precision-based content-overlap metric proposed by Papineni et al. [37] in 2002. The precision-based metric means that we compare the two texts by counting the number of words or n -grams in the generated text that occur in the human reference and dividing the count by the length of the reference text. Compared to ROUGE, BLEU focuses on precision and is suitable for tasks that favor high precision. However, recall is also essential for our task since we expect more words from the expert feedback to appear in the system-generated feedback. On the other hand, both recall and precision can be taken into account for the ROUGE metric [7].

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) is the other most commonly used content-overlap metric introduced by Lin [25] in 2004. The original ROUGE

score is basically a recall-based version of BLEU. That is, for ROUGE, we check how many words or n -grams in the reference text appear in the generated text. Nevertheless, entirely removing precision can have substantial adverse effects (e.g., a system may generate extremely long text strings to capture all words in the reference text). In recent years, ROUGE has commonly referred to ROUGE-F₁ that combines both precision and recall ROUGE scores in the harmonic mean. We report ROUGE-F₁ scores in results since both recall and precision are vital for our system.

Model-based metrics: Model-based metrics use learned representations of words and sentences to compute semantic similarity between generated and reference texts. Model-based metrics are generally more correlated with human judgment than content-overlap metrics, but their behavior is not interpretable [9]. Given the excellent performance of BERT [11] across many tasks, recent work on model-based metrics commonly uses pre-trained contextual embeddings from BERT or similar pre-trained language models for evaluating the semantic equivalence between the texts [55, 53, 43]. One of the most widely used metrics is BERTScore [55].

BERTScore is a model-based metric proposed by Zhang et al. in 2020 and has been shown to correlate well with human judgments for many text-generation tasks. BERTScore leverages the pre-trained embedding from BERT and matches words in generated and reference sentences by cosine similarity. Moreover, BERTScore considers both precision and recall and combines them to compute an F₁ measure, which is appropriate for evaluating generated feedback in our task. Thus, we employ BERTScore to measure semantic similarity between expert feedback and generated feedback.

Human-centered evaluation: Human-centered evaluation asks human evaluators to judge the quality of generated text along some specific dimensions (e.g., readability). Caligiuri and Thomas [6] found that a positive tone and suggestions for improvement are key features of good reviews. Jia et al. [20] mentioned that providing suggestions, mentioning problems, and using a positive tone, are main characteristics of effective feedback. Celikeyilmaz et al. [9] pointed out that fluency and factuality are essential for evaluating system-generated text. In this paper, we manually evaluate generated feedback in five dimensions: Readability, Suggestions, Problems, Positive Tone, and Factuality. The details of these evaluation dimensions are described in Section 4.5.3.

2.3 Ethical Concerns in Feedback Generation

To date, ethical issues in feedback generation have received scant attention in the research literature. Celikeyilmaz et al. [9] pointed out that a potential ethical issue that may appear in text-generation tasks is the problem of generating improper or offensive language. Li et al. [24] also mentioned that using pre-trained language models for text-generation tasks may raise ethical issues, such as generating private content, because corpora used for pre-training are pulled from the web without carefully filtering. However, there is still a lack of systematic methods for evaluating how effectively a system can avoid generating inappropriate text. For these reasons and others, this work considers the ethical implications of an automated feedback system on student project reports by manually inspecting all generated feedback.

Table 1: Sample Human Reference Reviews

Very good writeup, as far as it goes. Good discussion of test cases and reasons for refactoring. It would have helped to see some code snippets. Good section on Future Refactoring Opportunities.
A very good description of changes, and appropriate code snippets are shown. Manual testing is shown with annotated screenshots, which are very useful. In a Rspec test, sending emails to an actual person’s address is not a good practice.
The wiki page covers all necessary items, but the “test plan” part can be more elaborated. And the last screenshot is useless. It will be better to show the DB records, instead of table structure.

Table 2: Statistics on the Dataset. We measure the percentage of reports that contain more than 1024 tokens since it is the maximum input length limit of the BART model, and we utilize BART to craft the generation function in our system.

# of data samples = 484		Average	Percentile					Maximum	Over 1024 tokens
			5th	25th	50th	75th	95th		
Reports	# of words	1193	405	734	1060	1499	2374	6512	
	# of subword tokens	1643	583	1040	1489	2025	3272	8422	76 %
Summarized Reports	# of words	704	395	649	742	803	865	908	
	# of subword tokens	951	580	1004	1021	1023	1024	1024	0 %
Reference Reviews	# of words	55	13	33	48	72	114	258	
	# of subword tokens	71	19	43	61	90	147	335	-

3. DATA

In this section, we introduce a new dataset collected for this study. Firstly, we describe the data source in Section 3.1. Then, we explain in Section 3.2 how participants’ privacy rights were respected during the data collection process. Finally, we present statistics on the dataset in Section 3.3.

3.1 Data Source

The data used in this study are collected from a graduate-level object-oriented development course at a public university in the United States. For a course assignment, two to four students form a group and work together on some course project, bid on topics from a list of potential projects provided by course instructors. The dataset includes group projects where students refactor code from an open-source project Expertiza, add new features to it, or write automated tests for a software module that needs them.

As a part of project deliverables, each team is required to submit a group report to document the work that has been completed, methodologies that they have utilized, and other project-related material (e.g., how they test their code). Such group reports are also called wiki pages since they are added to the wiki document maintained by the open-source project Expertiza. The instructor reviews each of these wiki pages (i.e., each group report) and provides textual feedback on each of them. To better understand our data, URLs to three anonymous and de-identified reports are provided in footnotes¹²³. Three randomly sampled⁴ instructor reviews from our dataset are displayed in Table 1.

¹Sample report 1: <https://anonymous.4open.science/r/EDM22-BF52/Student%20Report%201.pdf>

²Sample report 2: <https://anonymous.4open.science/r/EDM22-BF52/Student%20Report%202.pdf>

³Sample report 3: <https://anonymous.4open.science/r/EDM22-BF52/Student%20Report%203.pdf>

⁴According to our IRB protocol, in order to protect participants’ privacy rights, we do not display instructor reviews for the sample group reports.

3.2 Privacy protection

In this work, we take our responsibility to protect the privacy of students’ data very seriously. The use of the dataset has been approved by the IRB at our institution. Sensitive student data was de-identified and handled in a way that is FERPA compliant. More specifically, our data protection and de-identification procedure consist of four main steps: 1) we took our data from an anonymized database, which uses random identifiers for students and groups; 2) we utilized regular-expression techniques to automatically remove all names from reports; 3) we manually inspected and removed remaining sensitive data, such as links to documents that might identify individual authors; 4) we stored data securely on a cloud drive managed by the university.

3.3 Statistics on the Dataset

We collected de-identified students’ project reports and associated textual feedback provided by instructors from twelve semesters between Spring 2015 and Spring 2021. This gave us a set of 484 group projects. Table 2 summarizes statistics from the dataset. Note that since the pre-trained language model BART (detailed in Section 4) has a maximum input length limit of 1024 tokens⁵, we employ an unsupervised method to summarize original reports to lengths acceptable for input into the pre-trained BART. Statistics on these summarized reports are provided in the third row of the table.

We measured the average number of words and tokens for reports, summarized reports, and expert reviews in our dataset. The average number of words for each original report is 1193, which corresponds to 1643 subword tokens. We found that 75.8% of the reports comprise more than 1024 tokens. The average number of words in each summarized report is 704 (equivalent to 951 subword tokens). For expert reviews, the average number of words and the average number of tokens per instructor review are 55 and 71, respectively.

⁵A token is an instance of a sequence of characters that are grouped together as a useful semantic unit for processing. It is similar to, but not identical with, morpheme.

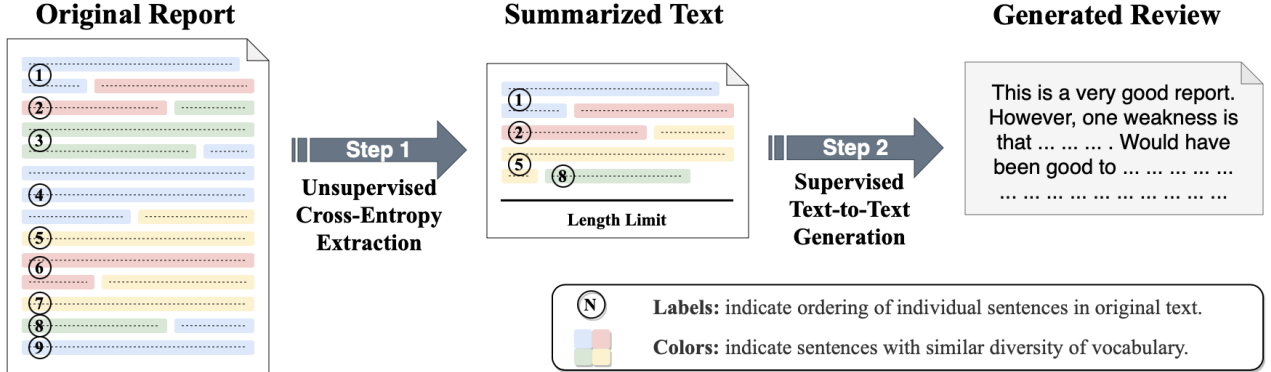


Figure 1: Operation of our *Insta-Reviewer* System. The system uses a select-then-generate paradigm [44, 18, 52]. The first step is to extract salient sentences (within the length limit) from students’ project reports. The second step is to utilize a supervised NLP model to generate feedback for students’ reports. Details of the system are described throughout Section 4.

4. METHODOLOGY

In this section, we detail our data-driven approach for automatically generating feedback on students’ project reports. We first formally define our task in Section 4.1. Then, we present the overall design of our feedback-generation system in Section 4.2. After that, Section 4.3-4.4 elaborates all components of the system. A set of metrics for evaluating generated textual feedback is given in Section 4.5. Finally, in Section 4.6, some ablation experiments are proposed to understand the contribution of each component.

4.1 Problem Formulation

We formulate the task of automatic feedback generation for students’ project reports as a *text-to-text generation* problem, where the source text is a long project report and the target text is a review. Our dataset can be represented as $D = (X^i, Y^i)_{i=1}^{N=484}$, where $X^i = \langle x_1^i, \dots, x_j^i, \dots, x_n^i \rangle$ denotes a sequence of input tokens representing an instance of report, and $Y^i = \langle y_1^i, \dots, y_k^i, \dots, y_m^i \rangle$ denotes a sequence of output tokens representing the corresponding textual feedback. Each token x_j^i or y_k^i is drawn from a token vocabulary \mathcal{V} .

Then, the problem can be formally described as:

$$Y = \mathcal{F}_{\mathcal{M}}(X, \mathbb{C}) \quad (1)$$

where the model, or generation function, $\mathcal{F}_{\mathcal{M}}$ takes a sequence of tokens X (i.e., a project report) as the input, and produces a sequence of tokens Y (i.e., generated feedback for the project report) as the output, while satisfying a set of constraints \mathbb{C} , which is a collection of desired properties (e.g., fluency, coherence, and length) for the output text.

The objective of the task is to effectively model the generation function $\mathcal{F}_{\mathcal{M}}$ in a data-driven manner using the dataset, so that it can generate plausible and readable feedback for unseen reports. In this work, the generation function $\mathcal{F}_{\mathcal{M}}$ is crafted based on a pre-trained language model (PLM) called BART (detailed in Section 4.4), which has been demonstrated to be the state-of-the-art method to model the generation function for various text-to-text generation tasks [23].

4.2 System Design

Despite the fact that the pre-trained language model BART is an effective method to model the generation function $\mathcal{F}_{\mathcal{M}}$, it has an input length limit of 1024 tokens⁶ [23]. Nevertheless, group reports are usually longer than that. In our dataset, 75.8% of the reports contain more than 1024 tokens, and the longest report is approximately eight times longer than that limit. One simple fix is to truncate the report by discarding all tokens beyond the length limit, but this can cause loss of critical information from the inputs (this hypothesis is verified in Section 4.6.1). Thus, we adopt a select-then-generate method [18, 44, 52] to generate feedback on student project reports, as illustrated in Figure 1.

Overall, the select-then-generate paradigm decomposes the problem into two sequential subproblems to resolve the issue of overlength input documents: 1) an unsupervised sentence-level extractive summarization task, and 2) a supervised PLM-based text-to-text generation task. More formally, the problem description becomes,

$$Y = \mathcal{F}_{\mathcal{M}}(\mathcal{S}_{\mathcal{E}}(X), \mathbb{C}) \quad (2)$$

where the input to the generation function $\mathcal{F}_{\mathcal{M}}$ becomes $\mathcal{S}_{\mathcal{E}}(X)$, which represents a summarized report. The new function $\mathcal{S}_{\mathcal{E}}$ represents an extractive summarizer, which can effectively extract salient sentences from an original report X and produce a summarized version of the report as the input to the feedback-generation function $\mathcal{F}_{\mathcal{M}}$.

Thus, our automated feedback-generation system comprises two stages. In the first stage, we use an unsupervised method, called cross-entropy extraction, to summarize original reports to lengths acceptable for input into the PLM BART (i.e., the implementation of the generation function $\mathcal{F}_{\mathcal{M}}$). In the second stage, we train the PLM BART on the summarized reports $\mathcal{S}_{\mathcal{E}}(X)$ and reference reviews Y . In the following sections (Section 4.3-4.4), we detail each stage of our feedback-generation system *Insta-Reviewer*.

⁶The length is limited to 1024 since the BART authors [23] chose this number and pre-trained the model with this limit.

4.3 Step 1: Cross-Entropy Extraction

The goal of the first step is to summarize original over-length reports to lengths acceptable for input into the pre-trained language model (PLM) BART that will be used in the second step, while retaining as diverse a subset of content as possible. We adopt an unsupervised extractive-summarization method, called cross-entropy extraction [15].

The cross-entropy method is an unsupervised technique that treats sentence-level extractive summarization as a combinatorial optimization problem [42]. More formally, we can let S_D denote the set of all sentences in the report we are trying to summarize. From this set we want to extract a subset of sentences $S \subset S_D$ that maximizes some quality target function $Q(S)$. This quality target function can be comprised of whatever features and measures deemed applicable to the task at hand. In our case, we used only a single feature in our quality function, namely the diversity of vocabulary in the summary. The reasoning behind this is that sentences with a more diverse vocabulary will also cover a more diverse set of information from the text. To measure this diversity explicitly, we calculate the unigram LM entropy of the summary S , as shown below,

$$Q(S) = - \sum_{w \in S} p_S(w) \log p_S(w) \quad (3)$$

$$p_S(w) = \frac{\text{COUNT}(w)}{\text{LEN}(S)} \quad (4)$$

Note that in the above equations w represents a single word in the summary S . Additionally, the function $\text{COUNT}(w)$ measures the frequency of the word w in summary S and $\text{LEN}(S)$ is the total number of words in the summary. Additionally, to encourage the method to prefer summaries as close to the length constraint as possible, we added an additional term to this quality function that is proportional to the number of tokens in S , denoted $\text{TOKENS}(S)$, the intuition behind this being that the BART model will perform better in most cases when it has more text to work with. The final quality function with this added length term is shown below, where β , our proportionality constant, is a hyper-parameter of the model.

$$Q(S) = \beta \cdot \text{TOKENS}(S) - \sum_{w \in S} p_S(w) \log p_S(w) \quad (5)$$

In order to enforce the length constraint on our summaries, we simply assign $Q(S) = -\infty$ whenever S has a length of greater than 1024 BART tokens. The actual output of the cross-entropy method is a vector $p = \langle p_1, p_2, \dots, p_n \rangle$ indicating the probability of selection in the summary for each of the n sentences in the original document. Initially, these probabilities start out the same for each sentence, but they quickly convert to either 0, for sentences that result in low Q values, or to 1, for sentences that result in high Q values. Below are the steps of the algorithm we used for carrying out this cross-entropy extraction procedure.

1. **Preprocessing Text:** For each student project report, we split it into its n sentences, enumerating each according to their order in the report. We then tokenize each sentence into word tokens, being sure to remove all stop words, punctuation, etc., when doing so.

2. **Initialize p :** Initially we want each sentence to be equally likely to be chosen, so set $p_0 = \langle 1/2, 1/2, \dots, 1/2 \rangle$. If $n > 60$ then we reduce this probability to ensure a sufficient sample of summaries that meet our length constraint. Then we set $t = 0$.

3. **Sample Summaries:** We sample N Bernoulli vectors, X_1, X_2, \dots, X_N according to the probability vector p_{t-1} . The sentences selected from each of these vectors define our N sample summaries S_1, S_2, \dots, S_N . Set $t = t+1$.

4. **Quality Scores:** For each of the summaries S_i , we calculate its quality performance score according to the above equations. We determine the cutoff value of the elite sample, γ_t , which is the Q value of the $(1 - \rho)$ sample quantile.

$$\gamma_t = Q(S)_{[(1-\rho)N]} \quad (6)$$

5. **Update p :** We use the sample values to update our probabilities, storing them as \hat{p}_t , according to the update rule below:

$$\hat{p}_{t,j} = \frac{\sum_{j=1}^N \delta_{[Q(S_j) \geq \gamma_t]} \delta_{[X_{i,j}=1]}}{\sum_{j=1}^N \delta_{[Q(S_j) \geq \gamma_t]}} \quad (7)$$

where $\delta_{[c]}$ is the Kronecker-delta function which evaluates to 1 if the condition c is satisfied, otherwise 0.

6. **Smooth p :** To balance exploration and exploitation of the summary samples, we smooth p_t like so:

$$p_t = \alpha \hat{p}_t + (1 - \alpha) p_{t-1} \quad (8)$$

7. **Termination:** If the value of γ_t has not changed in 3 iterations then the process terminates, returning the current p_t . Otherwise, it returns to step 3 and repeats.

In our implementation of the above algorithm, we found the following parameter settings to be optimal: $N = 1000$, $\rho = .05$, $\alpha = .7$. To get our final truncated summary text, we simply sample one more Bernoulli vector, X , using the final sentence extraction probabilities p . After doing so, we check that the resulting summary defined by the sentences in X meets our length constraint. If it does not, then we would sample a new Bernoulli vector X until we found a summary that did (though this was never necessary).

4.4 Step 2: PLM-based Generation Model

We now describe the second step of the approach. The objective of the second step is to effectively craft the feedback-generation function \mathcal{F}_M in Equation 2. We first introduce the BART model used for crafting \mathcal{F}_M , then we describe the beam-search method for improving the performance.

4.4.1 Modeling the Generation Function with BART

In this work, we employ a state-of-the-art PLM BART [23], which stands for bidirectional and auto-regressive transformers, to model the generation function \mathcal{F}_M . The BART model is suitable for text-to-text generation tasks since it utilizes an encoder-decoder architecture (as illustrated in Figure 2), which can effectively model complex mappings (i.e., the underlying patterns) from one sequence of text (e.g., summarized reports) to another (e.g., feedback).

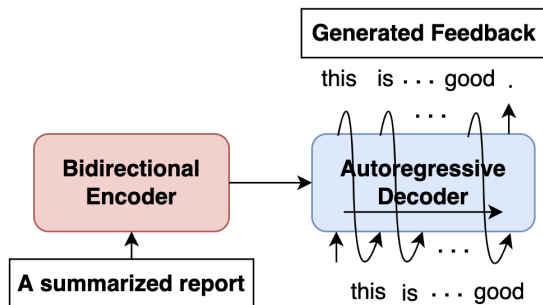


Figure 2: Illustration of an encoder-decoder architecture. The encoder can convert an input sequence of text (e.g., a summarized report) into a rich numerical representation, and then the decoder generates the output sequence (e.g., feedback) by iteratively predicting the most probable next word.

The BART framework consists of two steps: pre-training and fine-tuning. Instead of training the model from scratch on our dataset, the BART model is first pre-trained on a large generic corpus over different pre-training tasks, and then all parameters of the model are fine-tuned on our data. The model can acquire a sophisticated “understanding” of human grammar through the pre-training process, thus significantly reducing the need for annotated data when training the feedback-generation model while improving convergence rate and generalization [13]. In this work, we use the pre-trained checkpoint “facebook/bart-large-cnn⁷” to initialize all parameters of the model and then fine-tune the model on a training set drawn from our dataset D .

4.4.2 Diverse beam search for decoding

After fine-tuning the BART model on our data, we use the diverse beam search (DBS) [46] algorithm to decode the output sequences in inference time to generate better feedback. In the original BART model setting, feedback is generated by iteratively selecting the word with the highest probability at each position in the sequence, which is referred to as greedy decoding. Greedily choosing the word with the highest probability at each step might be optimal at the current spot in the sequence, but as we move through the rest of the full sentence, it might turn out to be a non-optimal choice (output can be ungrammatical, unnatural, and nonsensical) since the greedy decoding algorithm lacks backtracking.

On the other hand, the DBS algorithm keeps track of the top- n most probable next words, where n is the number of beams. The next set of beams is chosen by considering all possible next-word extensions of the existing set and selecting the n most likely extensions. Additionally, in order to improve the diversity in the outputs, all beams are divided into groups, and diversity between the groups is enforced by the DBS algorithm. In our experiments, we set beam size to 4 and the number of groups to 2 since this combination yields the best results. DBS is applied to all models in this work, including the BigBirdPegasus model (Section 4.6.2).

4.5 Evaluation Metrics

⁷<https://huggingface.co/facebook/bart-large-cnn>

We evaluate generated feedback with a comprehensive set of metrics, including a content-overlap metric ROUGE, a model-based metric BERTScore, and a new human-evaluation metric. As previously mentioned, the ultimate goal of Insta-Reviewer is to generate feedback that is helpful to students instead of generating the same feedback as provided by instructors. To this end, human-centered evaluation is considered the gold standard. ROUGE and BERTScore are employed to validate our human-evaluation results since human evaluations may be inconsistent and subjective, which can lead to erroneous conclusions [9]. The intuition is that while instructor feedback may only focus on certain aspects and be imperfect, it is valuable to know how similar the generated feedback is to the feedback provided by instructors. The implementations of the metrics are described below.

4.5.1 Content-overlap Metric: ROUGE

We use the standard ROUGE metric to measure content overlap between generated feedback and expert feedback. Specifically, we report the F_1 scores for ROUGE-1, ROUGE-2, and ROUGE-Lsum, respectively measuring the word-overlap, bigram-overlap, and longest common sequence between the texts. We obtain our ROUGE scores using the Google rouge package⁸ [25, 16]. Porter stemming is enabled to remove plurals and word suffixes (e.g., “ing”, “ion”, “ment”).

4.5.2 Model-based Metric: BERTScore

The BERTScore metric is leveraged to assess the semantic equivalence between generated feedback and expert feedback. We report the F1 measure of BERTScore that combines both precision and recall, which is proper for evaluating generated feedback in our task. We calculate the BERTScore utilizing the official BERTScore script⁹ [55].

4.5.3 Human Evaluation

After reviewing relevant papers (e.g., [6, 20, 50, 57]) and discussions among the authors of this paper, we selected the following five dimensions to evaluate the feedback manually: Readability, Suggestions, Problems, Positive Tone, and Factuality. Our scores for these five manual-evaluation dimensions are calculated as follows:

- (i) **Readability** (READ): In this work, readability is defined as the quality of feedback in grammar, word choice, and coherence. We judge it using a five-point rating scale: 0. Incomprehensible 1. Not fluent and incoherent 2. Somewhat fluent but incoherent 3. Fluent but somewhat incoherent 4. Fluent and coherent.
- (ii) **Suggestions** (SUGG): Providing suggestions is a key feature of quality feedback that is valuable to students. We give a score of 1 if the feedback contains at least one valid suggestion statement that can guide the reviewee in how to correct a problem or make improvements. Otherwise, we give a score of 0.
- (iii) **Problems** (PROB): Pointing out something that is going wrong in students’ work is also important for helping learners. We give a score of 1 if the feedback describes at least one issue that needs to be addressed in the student report. Otherwise, we give a score of 0.

⁸<https://github.com/google-research/google-research/tree/master/rouge>

⁹https://github.com/Tiiiger/bert_score

- (iv) **Positive Tone** (TONE): Feedback phrased in a positive tone can better stimulate students’ reflective competence. We give feedback a score of 1 if it has an overall positive semantic orientation, 0.5 if it is neutral, and 0 if it is negative.
- (v) **Factuality** (FACT): The statements in generated feedback should be factually correct. Otherwise, they may inadvertently mislead the reviewee and negatively impact learning. Factuality is calculated as:

$$\text{FACT} = \frac{\text{Count}(\text{factually correct statements})}{\text{Count}(\text{total statements})}$$

where the numerator is the total number of factually correct statements, and the denominator is the total number of statements in the feedback. If the denominator is 0, we directly give a score of 0.

This set of evaluation criteria is certainly not perfect, but it balances the accuracy and cost of the evaluation. For example, we could score the “Problems” dimension in a more sophisticated and accurate way, but it would be very time-consuming and expensive. We leave more accurate and efficient human-evaluation criteria for future work.

4.6 Ablation experiments

In order to understand the contribution of each step in our method, we designed two ablation experiments:

4.6.1 Ablation Exp. 1 - Naïve BART:

In the first ablation experiment, we intend to understand the contribution of the cross-entropy (CE) summarization. As we mentioned in Section 4.2, a straightforward way to solve the BART’s length-limit problem is to truncate all tokens beyond the length limit, which we hypothesize may lead to a loss of critical information from the inputs. Therefore, we measure the performance of a BART model that simply truncates all tokens exceeding the length limit of 1024 as input, and we call this setup - “Naïve BART.” If adequate information is contained in the truncated reports to generate feedback on students’ reports, then “CE + BART” should perform similarly to “Naïve BART.” Otherwise, it would demonstrate that using CE to summarize the original students’ project reports is better than simply truncating all tokens that exceed the length limit. In other words, it will suggest that the CE method can effectively summarize students’ reports while retaining critical information that helps the model generate feedback, and the CE step is necessary.

4.6.2 Ablation Exp. 2 - BigBirdPegasus:

In the second ablation experiment, we investigate whether the “CE + BART” method can be replaced using the recently proposed sparse-attention-based PLM BigbirdPegasus [54], which extends the input length limit to 4096 tokens. Briefly, BigBirdPegasus increases the input length limit at the cost of using a sparse attention mechanism, which may reduce the ability of the model to fit complex mappings between texts. Similar to BART, BigBirdPegasus also consists of the pre-training and fine-tuning steps. In this experiment, we use the pre-trained checkpoint “google/bigbird-pegasus-large-arxiv¹⁰” to initialize all parameters of BigbirdPegasus and then fine-tune the model on our data.

¹⁰<https://huggingface.co/google/bigbird-pegasus-large-arxiv>

5. EXPERIMENTS AND RESULTS

5.1 Experimental Setup

5.1.1 Training Details

For all experiments, we train our models with a batch size of 1/2, a learning rate of 2e-5/3e-5/5e-5, epochs of 2/3, and the AdamW optimizer [26] with a weight decay of 0.01 and a linear rate scheduler of 10% warm-up steps. For our dataset D , we use an 80-10-10 split for training, validation, and test data. After finding the optimal hyper-parameters, we merge the training and validation sets into the new training data.

5.1.2 Hardware Setup

The BART models are trained on an NVIDIA RTX6000 GPU (24GB). The BigBirdPegasus model (mentioned in Section 4.6.2) is trained on an NVIDIA A6000 GPU (48GB). We also employ the automatic mixed-precision training (use of both 16-bit and 32-bit floating-point types) to speed up the training processes.

5.2 Experimental Results

To investigate the potential promise of our system, we design experiments to answer the following research questions:

RQ1: How effective is the proposed approach for generating feedback on students’ project reports?

The goal of our first experiment is to find out if our automated feedback system Insta-Reviewer is effective in generating some readable and helpful feedback for students’ project reports. The evaluation results of Insta-Reviewer are shown in Table 3. Some generated feedback for actual student reports is shown in Table 5, and their evaluation scores are shown in Table 4. According to Table 3, the ROUGE1, ROUGE2, ROUGESum, and BERTScore for our “CE + BART” method are 28.54, 6.39, 18.21, and 59.18, respectively. These results imply that the generated and expert feedback are basically consistent in semantics, and there is some overlap in words (more precisely, n -grams).

We also evaluated the feedback provided by instructors, and the first row shows the human-evaluation scores for it. Compared to the expert feedback, we surprisingly find that in terms of “Problems” and “Positive Tone,” our method can outperform human experts by 6% and 2%. However, it is worth noting that the generated feedback tends to mention more generic problems (e.g., missing a test plan) rather than project-specific issues (e.g., “xxx files should be described”). Additionally, the expert feedback can outperform the generated feedback, with gaps of 6%, 16%, 15.2% for “Readability,” “Suggestions,” and “Factuality,” respectively. In summary, although our system is not as good as experts at providing suggestions and may produce some non-factual statements, it is good at generating fluent and positive feedback that mentions problems that need to be addressed.

RQ2: What are the problems of system-generated reviews? What are they not good at?

In the second experiment, we aim to explore the potential issues in system-generated feedback in more detail. Although the vast majority of the feedback is fluent, positive, and factually correct, we find two potential problems after manually inspecting all system-generated feedback in the test set.

Table 3: ROUGE F₁ (with porter stemming), BERTScore, and Human Evaluation scores (%) on the test set (n=50). All our ROUGE scores have a 95% confidence interval (CI) of at most ± 2.98 as reported by the official ROUGE script (bootstrap resampling). Our BERTScore has a 95% CI of at most ± 1.54 . The BERTScore script does not support BigBirdPegasus currently.

Method	ROUGE			BS	Human Evaluation						
	R1	R2	RLsum		READ	SUGG	PROB	TONE	FACT	Average	
1. Expert Feedback	-	-	-	-	100.0	82.0	90.0	93.0	100.0	93.00	
Insta-Reviewer	2. CE + BART	28.54	6.39	18.21	59.18	94.0	66.0	96.0	95.0	84.8	87.16
Ablation Exp 1	3. Naïve BART	25.88	4.98	17.23	58.03	82.0	58.0	94.0	93.0	76.8	80.76
Ablation Exp 2	4. BigBirdPegasus	15.92	4.04	13.45	-	56.0	2.0	12.0	58.0	23.4	30.28

Table 4: ROUGE F₁ (with porter stemming), BERTScore, and Human Evaluation scores (%) for the sample generated feedback shown in the Table 5. †“I would suggest that the reader read the ...” is not a valid suggestion for the reviewee.

	ROUGE			BERTScore	Human Evaluation					
	R1	R2	RLsum		READ	SUGG	PROB	TONE	FACT	Average
Generated Feedback 1	27.52	1.87	20.18	59.17	100.0	100.0	100.0	100.0	100.0	100.0
Generated Feedback 2	45.09	18.00	21.57	66.79	100.0	0.00†	100.0	100.0	83.33	76.67
Generated Feedback 3	22.22	0.00	13.89	52.92	80.00	0.00	100.0	100.0	80.00	72.00

Table 5: Sample expert and generated feedback. Scores for the generated feedback are shown in Table 4. The generated feedback may contain **non-factual or **ambiguous** statements.**

Expert Feedback 1: “The page is pretty well organized in what changes they made. Their pictures were not taken as a screenshot, instead were just taken with their phone camera. They didn’t provide a manual test plan, which is problematic as we have no good way of verifying their changes without significant work involved.”

Generated Feedback 1: “The writeup is quite readable, and the changes are described in good detail. However, the code changes are not described in prose, and there is no explanation of how they are implemented. The test plan is not described, and it would be much more useful to have a description of the tests that are to be added.”

Expert Feedback 2: “Your examples say that code was added, but don’t describe the code or what it is used for. The reader needs to figure that out by reading the code. For someone following on, it would be much more useful to have a description of how the code works.”

Generated Feedback 2: “The writeup is very readable, but it does not explain the changes made to the code. It is not clear how the code was changed. The test plan is not described. There is no description of the tests. **I would suggest that the reader read the code** to figure out what they test.”

Expert Feedback 3: “Well written, but does not mention design patterns or principles. ‘Gemfiles’ should be downgraded. Schema changes should be shown as migrations and not direct changes in schema file.”

Generated Feedback 3: “The wiki page is very short, but **the changes are described well**. It is not clear why the changes were made. The code changes are not described in prose. **They** (\leftarrow code snippets) are just shown. I would have liked to see more explanation of the changes”

Firstly, the automated feedback system may occasionally ($\approx 15.2\%$ of all statements) generate some non-factual or ambiguous statements in the feedback. For instance, the “generated feedback 2” shown in Table 5 contains a non-factual statement (italicized): “... there is no description of the tests. *I would suggest that the reader read the code to figure out what they test.*” However, this statement should probably be “... a description of the tests needs to be added. Otherwise, readers need to read the code to figure out what you test.”

Secondly, some pieces of text frequently appear in different generated feedback. For instance, we find that 66% of the system-generated feedback contains the sentence “the writeup is very/quite readable.” We speculate that this happens because 14% of the expert feedback that used for training contains the exact same sentence, which introduces some sort of “imbalance” problem. This repetition problem is not necessarily a drawback of the system, but it suggests that high-frequency text pieces can influence the generation.

RQ3: How does the system perform compared with other potential methods?

We now turn our attention to the experimental results of the two ablation experiments detailed in Section 4.6. The evaluation results are presented in Table 3. The second row, the third row, and the fourth row show the performance scores for our Insta-Reviewer system, naïve BART (the method utilized for the ablation experiment 1), and BigBirdPegasus (the model used for the ablation experiment 2), respectively.

The goal of the first ablation experiment is to verify our hypothesis that using the cross-entropy extraction (CE) to summarize the original students’ project reports is better than simply truncating all tokens beyond the length limit of BART. By looking at the ROUGE scores and BERTScore, we can find that the “CE + BART” method consistently outperforms the “naïve BART” method, with gaps of 2.66, 1.41, 0.98, and 1.15 for R1, R2, RLsum, and BERTScore

Table 6: ROUGE F₁ (with porter stemming) and BERTScore on the test set (n=50) in small-data settings (≤ 100 samples). For each score, we report a 95% confidence interval (CI).

data points	ROUGE			BERTScore
	R1	R2	RLsum	
434	28.54±2.98	6.39±1.56	18.21±1.83	59.18±1.54
50	24.29±2.47	4.99±0.94	17.42±1.70	56.12±1.42
100	25.75±2.65	5.56±1.15	17.86±1.61	57.36±1.43

respectively. Additionally, based on the human evaluation scores, “CE + BART” significantly ($\geq 8\%$) outperforms the “Naïve BART” method on “Readability,” “Suggestions,” and “Factuality.” The “CE + BART” method can also achieve higher ($\geq 2\%$) scores on dimensions “problems” and “positive tone.” The results demonstrate that the CE method can effectively summarize students’ reports while retaining critical information that helps the model generate feedback.

The second ablation experiment aims to know whether the sparse-attention-based PLM BigBirdPeagsus can effectively fit the complex mappings from student reports to textual feedback and replace our “CE + BART” approach. As shown in Table 3, the results clearly indicate that “CE + BART” substantially outperform the BigBirdPeagsus model on all metrics. The results suggest that although the BigBirdPeagsus model extends the input length limit to 4096 by using the sparse-attention mechanism, it may not be suitable for complicated tasks such as generating feedback.

RQ4: How does the automated feedback system perform in different small-data settings?

In order to help other researchers who intend to apply our approach on their datasets, we evaluated the effectiveness of the approach in different small-data settings. Table 6 shows the ROUGE scores and BERTScore when training with only 50 and 100 data samples, respectively. The scores (R1=25.75, R2=5.56, RLsum=17.86, BS=57.36) for our model when training with 100 samples are similar to the scores (R1=25.88, R2=4.98, RLsum=17.23, BS=58.03) for the “Naïve BART” when training with 434 samples. The results demonstrate that the performance of our approach is acceptable in small-data settings. Thus, our method can potentially be applied to other tasks that have limited data.

RQ5: Does the Insta-Reviewer automatic feedback system raise any ethical concerns?

The objective of the last experiment is to investigate whether the automated feedback system raises any ethical concerns. In this work, we consider two main potential ethical issues related to the system. The first major concern is whether the system will generate improper or offensive language. The second potential issue is whether the generated feedback will contain some private content. Although we have filtered out all inappropriate information from our dataset, these ethical concerns may still arise since the BART model is pre-trained on a large-scale corpus crawled from the Internet without fine-grained filtering. Therefore, we manually vetted all generated feedback, and failed to find any of the aforementioned ethical transgressions.

6. CONCLUSION

Timely feedback is critical to learning because it is more likely to motivate students to stay on task and achieve their learning goals. This suggests that future AI-powered educational applications will include automated feedback systems to generate real-time feedback. In this paper, we have presented a data-driven system, named *Insta-Reviewer*, for generating instant textual feedback on students’ project reports. The system leverages a select-then-generate paradigm consisting of two main steps: 1) cross-entropy extraction and 2) BART-based supervised text-to-text generation. The results demonstrate that the generated feedback could achieve near-human performance and even outperform human experts in the “Problems” and “Positive Tone” dimensions. However, the system may occasionally generate some non-factual or ambiguous statements in the feedback. The generated feedback seems to be free of any ethical complications. Our work demonstrates the feasibility of automatic feedback generation for students’ project reports while laying the groundwork for future research on this topic.

Limitations: There are three main limitations to this study. Firstly, we simply used textual information extracted from the student reports and ignored all images. As a result, our model could not produce feedback like “Their pictures were not taken as a screenshot, instead were just taken with their phone camera.” If we could design a multi-modality model that incorporates all text, images, and artifacts such as code into the input, we would be able to provide more comprehensive feedback to students. Secondly, we used a set of metrics, including ROUGE, BERTScore, and human-evaluation scores to evaluate the generated feedback. However, ROUGE and BERTScore cannot accurately reflect the quality of the generated feedback. Human evaluation can more accurately assess the quality, but it is inconsistent, subjective, and time-consuming. Thus, we believe it is worthwhile to explore more effective automatic metrics to evaluate the generation. Thirdly, we manually inspected all generated feedback, and found that it did not raise any ethical concerns. Nevertheless, this result does not guarantee that the model will never produce improper or offensive language. Systematic methods should be investigated to evaluate how the system can avoid generating inappropriate language. However, this problem is particularly challenging because the output of neural networks is not always predictable.

Future Work: An important direction for future work is to investigate how to avoid generating non-factual statements in feedback. This problem of factual correctness has two potential solutions. The first promising way is to automatically evaluate the correctness of each sentence in generated feedback and remove all non-factual statements before delivering the feedback to students. Dusek & Kasner [12] have proposed an entailment-based model to evaluate the correctness of the generated text. However, this approach does not capture which part of the generated text is non-factual. Future work can explore further along this direction. The other possible method to address the problem of non-factual statements is to design new architectures that can more effectively capture the complex mappings from student reports to feedback. The latter approach is significantly more challenging but may get to the root of the problem.

7. REFERENCES

- [1] W. Alharbi. E-feedback as a scaffolding teaching strategy in the online language classroom. *Journal of Educational Technology Systems*, 46(2):239–251, 2017.
- [2] M. Ariely, T. Nazaretsky, and G. Alexandron. First steps towards nlp-based formative feedback to improve scientific writing in hebrew. *International Educational Data Mining Society*, 2020.
- [3] S. Basu, C. Jacobs, and L. Vanderwende. Powergrading: a clustering approach to amplify human effort for short answer grading. *Transactions of the Association for Computational Linguistics*, 1:391–402, 2013.
- [4] M. Borowczak et al. Communication in stem education: A non-intrusive method for assessment & k20 educator feedback. *Problems of Education in the 21st Century*, 65(1):18–27, 2015.
- [5] R.-M. Botarleanu, M. Dascalu, M.-D. Sirbu, S. A. Crossley, and S. Trausan-Matu. Readme—generating personalized feedback for essay writing using the readerbench framework. In *Conference on Smart Learning Ecosystems and Regional Development*, pages 133–145. Springer, 2018.
- [6] P. Caligiuri and D. C. Thomas. From the editors: How to write a high-quality review, 2013.
- [7] C. Callison-Burch, M. Osborne, and P. Koehn. Re-evaluating the role of bleu in machine translation research. In *11th conference of the european chapter of the association for computational linguistics*, pages 249–256, 2006.
- [8] D. Carless, D. Salter, M. Yang, and J. Lam. Developing sustainable feedback practices. *Studies in higher education*, 36(4):395–407, 2011.
- [9] A. Celikyilmaz, E. Clark, and J. Gao. Evaluation of text generation: A survey. *arXiv preprint arXiv:2006.14799*, 2020.
- [10] G. Deeva, D. Bogdanova, E. Serral, M. Snoeck, and J. De Weerd. A review of automated feedback systems for learners: Classification framework, challenges and opportunities. *Computers & Education*, 162:104094, 2021.
- [11] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [12] O. Dušek and Z. Kasner. Evaluating semantic accuracy of data-to-text generation with natural language inference. *arXiv preprint arXiv:2011.10819*, 2020.
- [13] D. Erhan, A. Courville, Y. Bengio, and P. Vincent. Why does unsupervised pre-training help deep learning? In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 201–208. JMLR Workshop and Conference Proceedings, 2010.
- [14] D. J. Evans, P. Zeun, and R. A. Stanier. Motivating student learning using a formative assessment journey. *Journal of anatomy*, 224(3):296–303, 2014.
- [15] G. Feigenblat, H. Roitman, O. Boni, and D. Konopnicki. Unsupervised query-focused multi-document summarization using the cross entropy method. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, page 961–964, New York, NY, USA, 2017. Association for Computing Machinery.
- [16] K. Ganesan. Rouge 2.0: Updated and improved measures for evaluation of summarization tasks. *arXiv preprint arXiv:1803.01937*, 2018.
- [17] E. F. Gehringer. A course as ecosystem: melding teaching, research, and practice. In *2020 ASEE Virtual Annual Conference Content Access*, 2020.
- [18] S. Gehrmann, Y. Deng, and A. M. Rush. Bottom-up abstractive summarization. *arXiv preprint arXiv:1808.10792*, 2018.
- [19] J. Hattie and H. Timperley. The power of feedback. *Review of educational research*, 77(1):81–112, 2007.
- [20] Q. Jia, J. Cui, Y. Xiao, C. Liu, P. Rashid, and E. F. Gehringer. All-in-one: Multi-task learning bert models for evaluating peer assessments. *arXiv preprint arXiv:2110.03895*, 2021.
- [21] J. A. Kulik and C.-L. C. Kulik. Timing of feedback and verbal learning. *Review of educational research*, 58(1):79–97, 1988.
- [22] S. Kusairi. A web-based formative feedback system development by utilizing isomorphic multiple choice items to support physics teaching and learning. *Journal of Technology and Science Education*, 10(1):117–126, 2020.
- [23] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.
- [24] J. Li, T. Tang, W. X. Zhao, J.-Y. Nie, and J.-R. Wen. A survey of pretrained language models based text generation. *arXiv preprint arXiv:2201.05273*, 2022.
- [25] C.-Y. Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.
- [26] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.
- [27] C. Lu and M. Cutumisu. Integrating deep learning into an automated feedback generation system for automated essay scoring. *International Educational Data Mining Society*, 2021.
- [28] A. Malik, M. Wu, V. Vasavada, J. Song, M. Coots, J. Mitchell, N. Goodman, and C. Piech. Generative grading: Near human-level accuracy for automated feedback on richly structured problems. *arXiv preprint arXiv:1905.09916*, 2019.
- [29] A. Mallavarapu. Exploration maps, beyond top scores: Designing formative feedback for open-ended problems. In *Proceedings of The 13th International Conference on Educational Data Mining*, 2020.
- [30] E. Mannix and M. A. Neale. What differences make a difference? the promise and reality of diverse teams in organizations. *Psychological science in the public interest*, 6(2):31–55, 2005.
- [31] S. Marwan, Y. Shi, I. Menezes, M. Chi, T. Barnes, and T. W. Price. Just a few expert constraints can help: Humanizing data-driven subgoal detection for

- novice programming. *International Educational Data Mining Society*, 2021.
- [32] P. J. Mensink and K. King. Student access of online feedback is modified by the availability of assessment marks, gender and academic performance. *British Journal of Educational Technology*, 51(1):10–22, 2020.
- [33] R. Nagata, M. Vilenius, and E. Whittaker. Correcting preposition errors in learner english using error case frames and feedback messages. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 754–764, 2014.
- [34] S. Narciss, S. Sosnovsky, L. Schnaubert, E. Andrès, A. Eichelmann, G. Gogvadze, and E. Melis. Exploring feedback and student characteristics relevant for personalizing feedback strategies. *Computers & Education*, 71:56–76, 2014.
- [35] A. M. Olney. Generating response-specific elaborated feedback using long-form neural question answering. In *Proceedings of the Eighth ACM Conference on Learning@ Scale*, pages 27–36, 2021.
- [36] J. W. Orr and N. Russell. Automatic assessment of the design quality of python programs with personalized feedback. *arXiv preprint arXiv:2106.01399*, 2021.
- [37] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.
- [38] A. Poulos and M. J. Mahony. Effectiveness of feedback: The students’ perspective. *Assessment & Evaluation in Higher Education*, 33(2):143–154, 2008.
- [39] P. Race. Using feedback to help students to learn. *The Higher Education Academy*, 2001.
- [40] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.
- [41] J. Roschelle, J. Lester, and J. Fusco. Ai and the future of learning: Expert panel report. *Digital Promise*, 2020.
- [42] R. Y. Rubinstein and D. P. Kroese. *The Cross Entropy Method: A Unified Approach To Combinatorial Optimization, Monte-Carlo Simulation (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2004.
- [43] T. Sellam, D. Das, and A. P. Parikh. Bleu: Learning robust metrics for text generation. *arXiv preprint arXiv:2004.04696*, 2020.
- [44] S. Subramanian, R. Li, J. Pilault, and C. Pal. On extractive and abstractive neural document summarization with transformer language models. *arXiv preprint arXiv:1909.03186*, 2019.
- [45] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
- [46] A. K. Vijayakumar, M. Cogswell, R. R. Selvaraju, Q. Sun, S. Lee, D. Crandall, and D. Batra. Diverse beam search: Decoding diverse solutions from neural sequence models. *arXiv preprint arXiv:1610.02424*, 2016.
- [47] W. Wang, G. Fraser, T. Barnes, C. Martens, and T. Price. Execution-trace-based feature engineering to enable formative feedback on visual, interactive programs. *feedback*, 32(1):2, 2021.
- [48] N. E. Winstone and D. Boud. The need to disentangle assessment and feedback in higher education. *Studies in Higher Education*, pages 1–12, 2020.
- [49] B. Woods, D. Adamson, S. Miel, and E. Mayfield. Formative essay feedback using predictive scoring models. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 2071–2080, 2017.
- [50] Y. Xiao, G. Zingle, Q. Jia, H. R. Shah, Y. Zhang, T. Li, M. Karovaliya, W. Zhao, Y. Song, J. Ji, et al. Detecting problem statements in peer assessments. *arXiv preprint arXiv:2006.04532*, 2020.
- [51] S. Young. Student views of effective online teaching in higher education. *The American Journal of Distance Education*, 20(2):65–77, 2006.
- [52] W. Yuan, P. Liu, and G. Neubig. Can we automate scientific reviewing?, 2021.
- [53] W. Yuan, G. Neubig, and P. Liu. Bartscore: Evaluating generated text as text generation. *Advances in Neural Information Processing Systems*, 34, 2021.
- [54] M. Zaheer, G. Guruganesh, A. Dubey, J. Ainslie, C. Alberti, S. Ontanon, P. Pham, A. Ravula, Q. Wang, L. Yang, and A. Ahmed. Big bird: Transformers for longer sequences, 2021.
- [55] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*, 2019.
- [56] R. Zhi, S. Marwan, Y. Dong, N. Lytle, T. W. Price, and T. Barnes. Toward data-driven example feedback for novice programming. *International Educational Data Mining Society*, 2019.
- [57] G. Zingle, B. Radhakrishnan, Y. Xiao, E. Gehringer, Z. Xiao, F. Pramudianto, G. Khurana, and A. Arnav. Detecting suggestions in peer assessments. *International Educational Data Mining Society*, 2019.

Sparse Factor Autoencoders for Item Response Theory

Benjamin Paaßen
German Research Center for
Artificial Intelligence
Berlin, Germany
benjamin.paassen@dfki.de

Malwina Dywel
Provadis Partner für Bildung
und Beratung GmbH
Frankfurt a.M., Germany

Melanie Fleckenstein
Provadis Partner für Bildung
und Beratung GmbH
Frankfurt a.M., Germany

Niels Pinkwart
Institute of Computer Science
Humboldt-University of Berlin
Berlin, Germany

ABSTRACT

Item response theory (IRT) is a popular method to infer student abilities and item difficulties from observed test responses. However, IRT struggles with two challenges: How to map items to skills if multiple skills are present? And how to infer the ability of new students that have not been part of the training data? Inspired by recent advances in variational autoencoders for IRT, we propose a novel method to tackle both challenges: The Sparse Factor Autoencoder (SparFAE). SparFAE maps from test responses to abilities via a linear operator and from abilities to test responses via an IRT model. All parameters of the model offer an interpretation and can be learned in an efficient manner. In experiments on synthetic and real data, we show that SparFAE is similar in accuracy to other autoencoder approaches while being faster to learn and more accurate in recovering item-skill associations.

Keywords

item response theory, logistic models, variational autoencoder, sparse factor analysis

1. INTRODUCTION

A foundational problem in educational data mining is to automatically infer students' ability from their observed responses in a test. Item response theory (IRT) addresses this problem by fitting a logistic model that describes how student ability and item difficulty interact to generate an observed response [5]. However, IRT faces at least two challenges. First, whenever a test involves multiple skills, we need to model the relation between skills and items, which standard IRT does not do [10]. Second, an IRT model contains student-specific parameters which are fitted to a specific population. For any new student, we need to fit at least one new parameter.

B. Paaßen, M. Dywel, M. Fleckenstein, and N. Pinkwart. Sparse factor autoencoders for item response theory. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 17–26, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853067>

The former challenge can be addressed via automatic methods for item-skill association learning, such as the Q -matrix method of Barnes et al. [2], the alternating least squares method [12], or sparse factor analysis [7]. The second challenge requires a student-independent parametrization of the model, which is offered by variants like performance factor analysis [11] or variational autoencoders [3]. In the present paper, we propose to address both challenges at once by combining sparse factor analysis with autoencoders, yielding a new method which we call sparse factor autoencoder (SparFAE).

In more detail, our contributions are: We introduce SparFAE, a sparse factor autoencoding method for IRT. We provide an interpretation for all parameters in the SparFAE model, as well as an efficient learning scheme. Further, we empirically compare SparFAE to sparse factor analysis [7] as well as variational autoencoders [16] on synthetic and real data and show that SparFAE is similar in accuracy to other encoders but is much faster to learn and more accurate in recovering item-skill associations. Finally, we use SparFAE to analyze an expert-designed math test and verify the identified Q -matrix against the expert-designed Q -matrix. The source code for all experiments can be found at <https://github.com/bpaassen/sparfae>.

2. BACKGROUND AND RELATED WORK

IRT models the responses of m students on a test with n items. In particular, let $y_{i,j}$ be a random variable, which is 1 if student i answered item j correctly and 0, otherwise. We assume that $y_{i,j}$ is Bernoulli-distributed, where the success probability is given as $p_{i,j} = \sigma(\theta_i - b_j)$, where $\sigma(x) = 1/(1 + \exp[-x])$ is the logistic link function, θ_i is an ability parameter for student i , and b_j is a difficulty parameter for item j [5]. The parameters θ_i and b_j need to be fitted to observed training data, for example, via likelihood maximization or Bayesian parameter estimation [1]. In particular, the negative log likelihood of the data (also known as crossentropy loss) is expressed by the formula

$$\ell = \sum_{i=1}^m \sum_{j=1}^n -y_{i,j} \cdot \log[p_{i,j}] - (1 - y_{i,j}) \cdot \log[1 - p_{i,j}]. \quad (1)$$

This loss is convex in the parameters θ_i and b_j , meaning that an optimal model can be found efficiently via nonlinear optimization algorithms.

Over the decades, numerous extensions of this basic scheme have been proposed, such as a discrimination parameter for each item (two-parameter model), a minimum probability of correct answers for each item (three-parameter model), partial credit, and hierarchical models [1, 4, 5]. In this paper, we care particularly about the extension to multiple underlying skills, sometimes called multidimensional IRT [10]. In such a model, we represent a student’s ability by a K -dimensional vector $\vec{\theta}_i$, where $\theta_{i,k}$ models the ability of student i for skill k . A consequence of including multiple skills is that we need to model the relationship between skills and items. In this paper, we assume a linear relationship that is captured by an $n \times K$ matrix \mathbf{Q} , where $q_{j,k}$ models how important skill k is to answer item j correctly. Overall, our model is described by the two equations:

$$p_{i,j} = \sigma(z_{i,j}) \quad \text{and} \quad \vec{z}_i = \mathbf{Q} \cdot \vec{\theta}_i - \vec{b}, \quad (2)$$

where \vec{z}_i is the vector of response logits for student i , and \vec{b} is the vector of all item difficulties.

Our setup begs the question: how to learn the matrix \mathbf{Q} ? Such coupling matrices between items and skills have been popularized by Tatsuoaka [13], who imposed $q_{j,k} = 1$ if skill k is required for item j and $q_{j,k} = 0$, otherwise. Traditionally, such \mathbf{Q} -matrices have been hand-designed by domain experts [9], but recently, automatic methods to learn \mathbf{Q} have emerged, such as the method of Barnes [2] or the alternating recursive method [12]. Crucially, finding an optimal binary \mathbf{Q} -matrix is challenging due to the discrete search space. To simplify the search, Lan et al. [7] have relaxed the problem by assuming continuous, non-negative entries of \mathbf{Q} and applying methods from sparse coding, resulting in a method called Sparse Factor Analysis (SPARFA).

SPARFA applies an alternating optimization scheme. First, we initialize student abilities $\vec{\theta}_i$ randomly, for example with Gaussian noise. Second, for each item j , we adapt the j th row of \mathbf{Q} and the difficulty b_j by solving the following optimization problem:

$$\begin{aligned} \min_{\vec{q}_j, b_j} \quad & \ell + \lambda_1 \cdot \|\vec{q}_j\|_1 + \lambda_2 \cdot (\|\vec{q}_j\|_2^2 + b_j^2) \\ \text{s.t.} \quad & q_{j,k} \geq 0 \quad \forall k \in \{1, \dots, K\}, \end{aligned} \quad (3)$$

where ℓ is the crossentropy loss (1), $\|\vec{q}_j\|_1 = \sum_{k=1}^K |q_{j,k}|$ is the 1-norm of \vec{q}_j , $\|\vec{q}_j\|_2^2 = \sum_{k=1}^K q_{j,k}^2$ is the squared Euclidean norm of \vec{q}_j , and λ_1 as well as λ_2 are hyperparameters of the method. The squared Euclidean norm is intended to regularize the model parameters with a Gaussian prior, as usual in IRT [1] (chapter 7). The 1-norm is motivated by sparse coding and encourages sparsity in \mathbf{Q} , meaning that the optimization process tends to find solutions where many of the entries in \mathbf{Q} are zero [17]. In other words, the model is encouraged to connect any item j only to a few skills instead of all skills. This is reminiscent of traditional \mathbf{Q} -matrices, where $q_{j,k}$ is only nonzero if skill k is required to answer item j correctly [13]. Finally, SPARFA enforces that no entry $q_{j,k}$ can become negative, because a negative $q_{j,k}$ would imply that a higher ability in skill k *reduces* my chance to answer item j correctly, which does not make sense [7]. Note that problem (3) is convex, such that it can be solved efficiently with nonlinear optimizers.

The third step of SPARFA is to optimize the ability parameters $\vec{\theta}_i$ for each student i . This is done by minimizing the crossentropy (1) plus a regularization term $\lambda_2 \cdot \sum_{k=1}^K \theta_{i,k}^2$. We now iterate steps two and three of the SPARFA algorithm until the parameters converge.

Just as in standard IRT, a challenge of SPARFA is that we can not immediately apply a learned model to new students. For every new student i , we need to fit new parameters $\vec{\theta}_i$. Many extensions of IRT have circumvented this problem by removing ability parameters altogether and only using item parameters. For example, performance factor analysis replaces the ability parameter by a weighted count of correct and wrong responses on past items for the same skill [11]. More recently, Converse et al. [3] proposed a variational autoencoder model to simplify the application of IRT models to new students.

A variational autoencoder (VAE) views the student abilities $\vec{\theta}_i$ as a compressed representation of the student’s response vector \vec{y}_i . More precisely, a VAE tries to learn an encoder function which compresses \vec{y}_i to abilities $\vec{\theta}_i$, and a decoder function which de-compresses $\vec{\theta}_i$ back into estimated responses \hat{y}_i , such that \vec{y}_i and \hat{y}_i are close and such that $\vec{\theta}_i$ is standard normal distributed [6]. As decoder, we use a multi-dimensional IRT model (2), whereas the encoder could be a multi-layer artificial neural network [3]. In contrast to traditional IRT models, a VAE model is typically non-convex and multi-layered, and thus needs to be optimized with deep learning methods [3, 6]. Wu et al. [16] have further extended the VAE version of IRT by analyzing the theory more closely and including the difficulty parameters \vec{b} as an additional input to the encoder. Fig. 1 illustrates the approach for a single-layer encoder. The encoder is given as $\vec{\theta}_i = \mathbf{A} \cdot \vec{y}_i + \mathbf{B} \cdot \vec{b} + \vec{\gamma}$ for some bias $\vec{\gamma}$ (Fig. 1, left, in orange), whereas the decoder is a multi-dimensional IRT model like in (2) (Fig. 1, right, in blue). Note that we obtain all models in this section as special cases of this diagram. If we set the connections \mathbf{B} to zero, we obtain the IRT-VAE of [3]. If we, further, remove the connections \mathbf{A} and treat $\vec{\theta}_i$ as parameters, we obtain SPARFA. Finally, if we set $K = 1$ and $q_{j,1} = 1$ for all j , we obtain a classic IRT model.

Interestingly, the state-of-the-art VAE approaches do not apply a sparsity penalty to facilitate interpretability of \mathbf{Q} . Further, deep learning can be quite slow. To address these limitations, we propose an autoencoder model based on the SPARFA loss, which we describe in the next section.

3. METHOD

Our proposed model is a single-layer autoencoder as illustrated in Fig. 1. More formally, our model can be concisely expressed in the following equations:

$$\begin{aligned} \vec{\theta}_i &= \mathbf{A} \cdot \vec{y}_i, \\ \vec{z}_i &= \mathbf{Q} \cdot \vec{\theta}_i - \vec{b}, \quad \text{and} \\ p_{i,j} &= 1 / (1 + \exp(-z_{i,j})), \end{aligned} \quad (4)$$

where the first equation expresses the encoder and the second and third equation the decoder.

Our interpretation of the parameters is as follows. \mathbf{A} maps

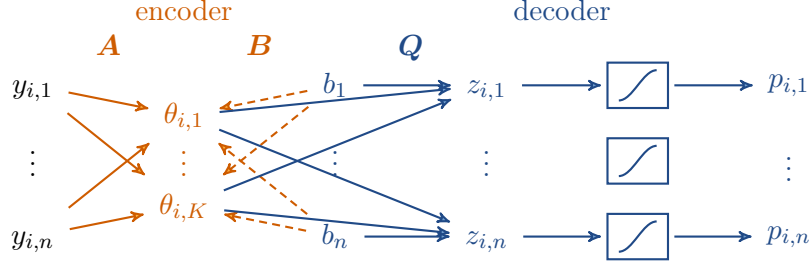


Figure 1: A sketch of a one-layer autoencoder for student responses, following Algorithm 1 of [16]. The Encoder is shown in orange (left), the decoder in blue (right). Encoder bias parameters are not shown for simplicity.

from responses to student ability, with $\alpha_{k,j}$ modeling the amount of ability in skill k that is expressed by answering item j correctly. Conversely, \mathbf{Q} maps from abilities to responses, with $q_{j,k}$ modeling how much skill k helps to answer item j correctly. b_j models the difficulty of item j , as before. Note that our model requires no student-specific parameters, such that it can be directly applied to new students.

Note that we do not include “backward” connections \mathbf{B} or encoder bias parameters $\vec{\gamma}$ in our model because they do not contribute to the model’s expressive power in the single-layer case. Consider a “full” model with $\vec{\theta}_i = \mathbf{A} \cdot \vec{y}_i + \mathbf{B} \cdot \vec{b} + \vec{\gamma}$. If we plug this expression into our equation for z_i , we obtain

$$\vec{z}_i = \mathbf{Q} \cdot (\mathbf{A} \cdot \vec{y}_i + \mathbf{B} \cdot \vec{b} + \vec{\gamma}) - \vec{b} = \mathbf{Q} \cdot \mathbf{A} \cdot \vec{y}_i + \mathbf{Q} \cdot \mathbf{B} \cdot \vec{b} + \mathbf{Q} \cdot \vec{\gamma} - \vec{b}.$$

We now absorb \mathbf{B} and $\vec{\gamma}$ into \vec{b} by re-defining \vec{b} as $\mathbf{Q} \cdot \mathbf{B} \cdot \vec{b} + \mathbf{Q} \cdot \vec{\gamma} - \vec{b}$, yielding Equations (4). Accordingly, our model requires only $2K + 1$ parameters per item.

We can train the parameters of our model by solving the following minimization problem, inspired by SPARFA.

$$\begin{aligned} \min_{\mathbf{A}, \mathbf{Q}, \vec{b}} \sum_{i=1}^m \sum_{j=1}^n -y_{i,j} \cdot \log[p_{i,j}] - (1 - y_{i,j}) \cdot \log[1 - p_{i,j}] \quad (5) \\ + \lambda_1 \cdot (\|\mathbf{A}\|_{1,1} + \|\mathbf{Q}\|_{1,1}) + \frac{\lambda_2}{2} \cdot (\|\mathbf{A}\|_F^2 + \|\mathbf{Q}\|_F^2 + \|\vec{b}\|^2) \\ \text{s.t. } \alpha_{k,j} \geq 0, q_{j,k} \geq 0 \quad \forall k \in \{1, \dots, K\}, j \in \{1, \dots, n\} \end{aligned}$$

where $\|\mathbf{A}\|_{1,1} = \sum_{j=1}^n \sum_{k=1}^K |\alpha_{k,j}|$ denotes the entry-wise 1-norm, and where $\|\mathbf{A}\|_F^2 = \sum_{j=1}^n \sum_{k=1}^K \alpha_{k,j}^2$ denotes the squared Frobenius norm. Since the resulting model is an autoencoder-variant of Sparse Factor Analysis, we call it Sparse Factor Autoencoder (SparFAE). We denote the objective function as ℓ_{SparFAE} . As in SPARFA, the Frobenius norm applies a Gaussian prior on the parameters, whereas the 1-norm encourages sparsity. We also apply the same non-negativity constraints as in SPARFA to ensure a meaningful interpretation of \mathbf{A} and \mathbf{Q} . Additionally, the non-negativity constraints are likely to further enhance sparsity, as indicated by non-negative matrix factorization [8].

In contrast to SPARFA, we can *not* decompose this problem into independent problems for each item because there are item-to-item-dependencies: Manipulating $\alpha_{k,j}$ also influences the abilities $\theta_{i,k}$, which in turn influence the probability $p_{i,j'}$ for any item j' with $q_{j',k} \neq 0$. Accordingly, we

need to perform a joint optimization of all parameters. However, we do not need to resort to deep learning. Instead, we propose a standard L-BFGS solver, as implemented in the minimize method of scipy [14]. This is facilitated by the surprisingly simple expression for the gradients:

$$\begin{aligned} \nabla_{\mathbf{A}} \ell_{\text{SparFAE}} &= \mathbf{Q}^T \cdot \Delta^T \cdot \mathbf{Y} + \lambda_1 \cdot \mathbf{1} + \lambda_2 \cdot \mathbf{A}, \\ \nabla_{\mathbf{Q}} \ell_{\text{SparFAE}} &= \Delta^T \cdot \mathbf{Y} \cdot \mathbf{A}^T + \lambda_1 \cdot \mathbf{1}^T + \lambda_2 \cdot \mathbf{Q}, \text{ and} \\ \nabla_{\vec{b}} \ell_{\text{SparFAE}} &= -\vec{1}^T \cdot \Delta + \lambda_2 \cdot \vec{b}, \end{aligned} \quad (6)$$

where \mathbf{Y} is the $m \times n$ matrix of all responses, where Δ is the $m \times n$ matrix with entries $\delta_{i,j} = p_{i,j} - y_{i,j}$, where $\mathbf{1}$ is the $K \times n$ matrix of only ones, and where $\vec{1}$ is an m -dimensional vector of ones. Regarding computational complexity, notice that the matrix products in (6) require $\min\{2 \cdot K \cdot m \cdot n, n^2 \cdot (m + K)\}$ operations, such that each optimization step is in $\mathcal{O}(m \cdot n)$ for constant K . We can simplify our optimization further by inspecting the relationship between \mathbf{A} and \mathbf{Q} .

3.1 Single Matrix Variant

Note that the matrices \mathbf{A} and \mathbf{Q} have related interpretations. Intuitively, if skill k helps more with item j (high $q_{j,k}$), we would also expect that answering item j correctly is an indicator for skill k (high $\alpha_{k,j}$). Accordingly, it stands to reason that $\mathbf{A} = \mathbf{Q}^T$.

We can also motivate this setting mathematically. In particular, $\mathbf{A} = \mathbf{Q}^T$ is optimal if \mathbf{Q} is orthogonal, meaning $\mathbf{Q}^T \cdot \mathbf{Q}$ equals the identity matrix \mathbf{I} . In that case, $\mathbf{Q} \cdot \mathbf{Q}^T \cdot \vec{y}_i$ is the orthogonal projection of \vec{y}_i onto the hyperplane spanned by \mathbf{Q} . In other words, $\mathbf{Q} \cdot \mathbf{Q}^T \cdot \vec{y}_i$ is the most similar point to \vec{y}_i we can achieve with the decoder \mathbf{Q} .

However, is it plausible that \mathbf{Q} is orthogonal? Indeed, $\mathbf{Q}^T \cdot \mathbf{Q}$ becomes a diagonal matrix (orthogonal up to scaling) if every item tests exactly one skill. Let J_k be the set of items which test skill k . Then, we obtain: $(\mathbf{Q}^T \cdot \mathbf{Q})_{k,l} = \sum_{j=1}^n q_{j,k} \cdot q_{j,l} = \sum_{j \in J_k} q_{j,k}^2$ along the diagonal and zero off the diagonal. In other words, the sparser \mathbf{Q} becomes, the closer $\mathbf{A} = \mathbf{Q}^T$ is to optimal.

When we plug $\mathbf{A} = \mathbf{Q}^T$ into problem (5), we obtain:

$$\begin{aligned} \min_{\mathbf{Q}, \vec{b}} \quad & \sum_{i=1}^m \sum_{j=1}^n -y_{i,j} \cdot \log[p_{i,j}] - (1 - y_{i,j}) \cdot \log[1 - p_{i,j}] \\ & + \lambda_1 \cdot \|\mathbf{Q}\|_{1,1} + \frac{\lambda_2}{2} \cdot (\|\mathbf{Q}\|_F^2 + \|\vec{b}\|^2) \\ \text{s.t.} \quad & q_{j,k} \geq 0 \quad \forall k \in \{1, \dots, K\}, j \in \{1, \dots, n\}, \end{aligned} \quad (7)$$

where $\vec{z}_i = \mathbf{Q} \cdot \mathbf{Q}^T \cdot \vec{y}_i - \vec{b}$. The gradient becomes:

$$\nabla_{\mathbf{Q}} \ell_{\text{SparFAE}} = (\mathbf{Y}^T \cdot \mathbf{\Delta} + \mathbf{\Delta}^T \cdot \mathbf{Y} + \lambda_2 \cdot \mathbf{I}) \cdot \mathbf{Q} + \lambda_1 \cdot \mathbf{1}^T.$$

This concludes our description of the proposed method.

4. EXPERIMENTS

In our experiments, we evaluate our proposed approach, Sparse Factor Autoencoder (SparFAE), on both synthetic and real-world data. We compare Sparse Factor Analysis (SPARFA) [7], Variational item response theory with a novel lower bound (VIBO) [16], the two-matrix version of SparFAE (SparFAE2), as well as the single-matrix version (SparFAE1). As optimizers, we used L-BFGS for SPARFA and both SparFAE versions, and an Adam optimizer with learning rate 0.005, 100 epochs, and minibatch size 16 for VIBO (these settings are as similar as possible to the original work of [16]). The experimental source code with all details is available at <https://github.com/bpaassen/sparfae>.

4.1 Synthetic Experiments

First, we consider synthetic data, which we sample from a multivariate IRT model with $K = 5$ skills, standard normally distributed abilities $\theta_{i,k}$, and standard normally distributed difficulties b_j . We introduce two different sampling conditions for \mathbf{Q} : A) We sample a unique skill k for each item j and set $q_{j,k} = 1$, whereas all other entries of \mathbf{Q} remain zero. B) We first sample a number of skills $K_j \in \{1, \dots, 5\}$ for each item j with probability $p(K_j) = \frac{6-K_j}{15}$. Then, we draw K_j skills k without replacement and uniform probability for item j and set $q_{j,k}$ to a uniform random number in the range $[0.5, 1]$.

As evaluation measures, we use the area under the receiver-operator-curve in predicting correct responses (AUC), the correlation between the learned difficulties b_j and the actual difficulties (r_b), the correlation between the learned abilities $\theta_{i,k}$ and the actual abilities (r_θ), fraction of agreeing nonzero entries between the learned \mathbf{Q} matrix and the true \mathbf{Q} -matrix (r_Q), the time needed for training, and the time needed for prediction on new students. Since the ordering of skills is undefined, we allow arbitrary permutations of the skills in the learned \mathbf{Q} -matrix before computing r_Q . In practice, we re-order the columns of \mathbf{Q} according to the `linear_sum_assignment` function of `scipy` with the ground-truth \mathbf{Q} matrix [14]. We evaluate all measures on a separate sample of $m' = 100$ new students. We repeat all experiments 10 times for each of the hyperparameter settings in Table 1.

First, we inspect the effect of hyperparameters for $m = 100$ students and $n = 20$ items. Fig. 2 shows, from left to right, how AUC, r_b , r_θ , and r_Q change for higher regularization in conditions A (top) and B (bottom). For AUC, we observe

Table 1: Hyperparameter settings considered in the experiments.

setting	λ_{VAE}	λ_1	λ_2
1	10^{-5}	10^{-3}	10^{-3}
2	10^{-4}	0.05	10^{-3}
3	10^{-3}	1	10^{-3}
4	0.01	0.05	0.05
5	0.1	1	0.05
6	1	1	1

a slight degradation of all methods for higher regularization, with a notable decline for VIBO at the last setting. r_b generally rises for higher regularization, with the exception of SparFAE1, which stays relatively stable around 0.5. r_θ appears stable across regularization and improves only for SPARFA. r_Q improves for all methods with higher regularization in condition A (top), and remains roughly stable in condition B (bottom). For the remaining synthetic experiments, we report the results using hyperparameter setting 6 for SparFAE2 and SparFAE1, and hyperparameter setting 5 for SPARFA and VIBO. These settings maximize r_b , r_θ , and r_Q while retaining high AUC.

Fig. 3 displays the performance measures for varying numbers of students. We observe that AUC, r_θ , and r_Q tend to slightly increase for more students across methods and conditions, with only slight deviances for small numbers of students. The most striking impact is on r_b , which increases for SparFAE1 and VIBO, but decreases for SPARFA and SparFAE2.

Fig. 4 displays the performance measures for varying numbers of items. Across methods, AUC decreases, whereas r_b and r_θ increase and r_Q remains roughly stable for higher number of items. The decrease in AUC is likely explained by the fact that the models need to compress the information of more items into the same number of skills, which is bound to decrease performance. Conversely, it becomes easier to tease apart the difficulty of each single item for a higher number of items per skill (hence the improvement in r_b). Further, the more items we have in a test, the more accurate we can estimate the underlying ability, which is reflected in better r_θ values.

Finally, Fig. 5 summarizes the effect of hyperparameter setting, number of students, and number of items on training time in logarithmic plots. We observe that stronger regularization reduces the training time for both SparFAE variants, whereas it stays roughly constant for VIBO and SPARFA. This is likely because training time for SPARFA and VIBO is driven by the repeated optimization steps over students, whereas the training time for SparFAE is dominated by a single optimization process. Hence, SparFAE profits more from the simpler loss surface offered by higher regularization. As one would expect, all methods scale roughly linearly with the number of students, SPARFA with roughly 18 ms per student, VIBO with roughly 6 ms per student, and both SparFAE variants with roughly 1.5 ms per student (refer to the gray dashed reference lines). For the num-

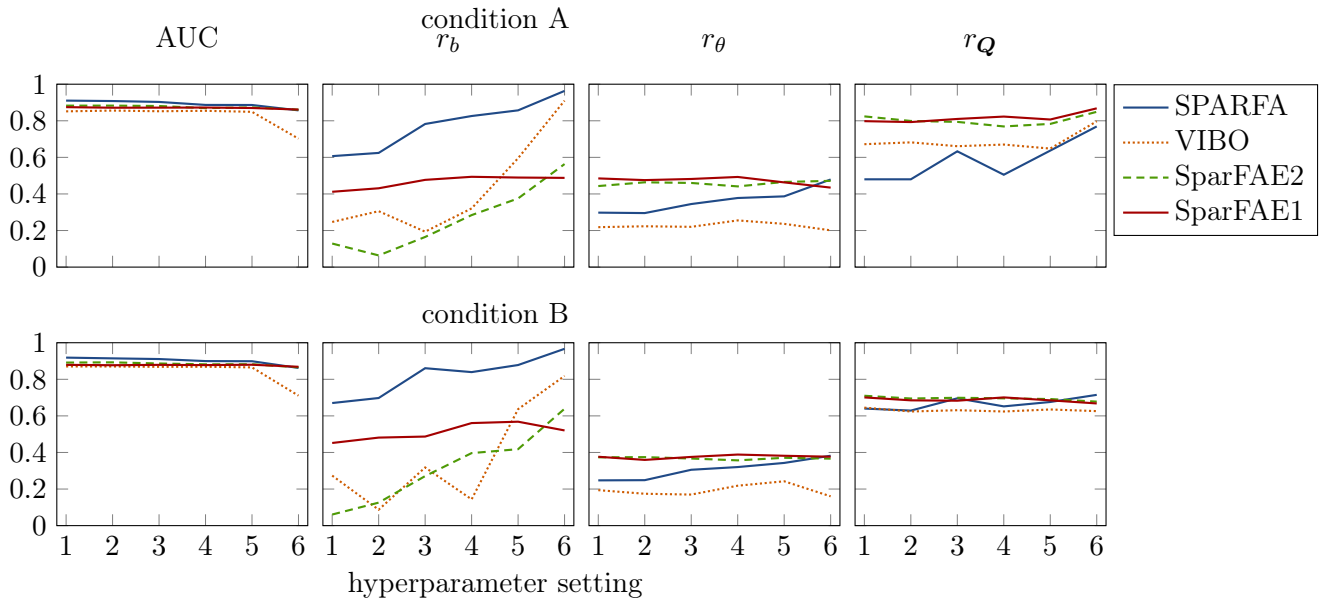


Figure 2: The effect of hyperparameter settings from Table 1 on various performance measures (from left to right) on the synthetic data, either with one skill per item (A, top), or multiple skills per item (B, bottom).

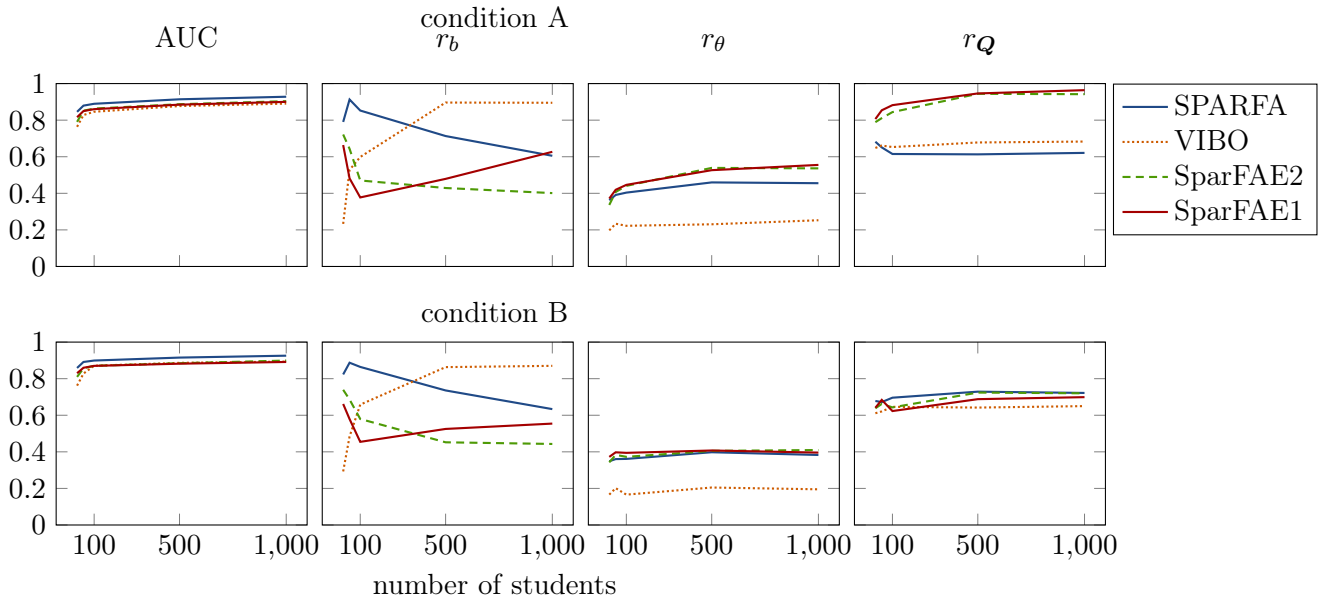


Figure 3: The effect of increasing the number of students on various performance measures (from left to right) on the synthetic data, either with one skill per item (A, top), or multiple skills per item (B, bottom).

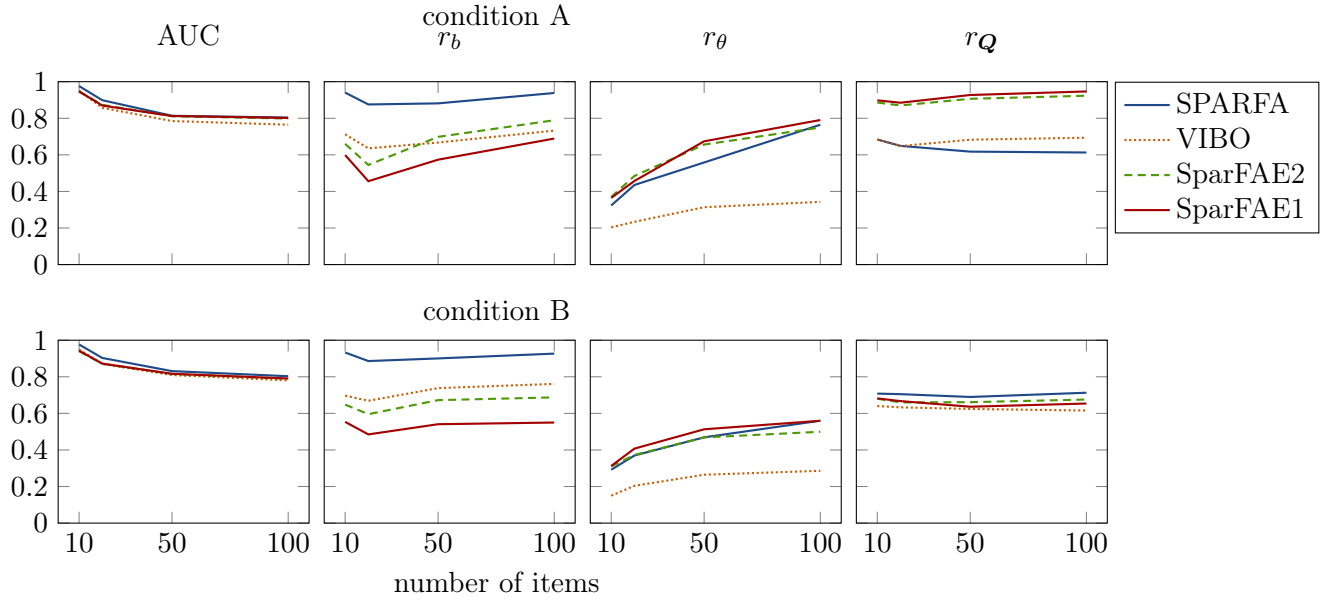


Figure 4: The effect of increasing the number of items on various performance measures (from left to right) on the synthetic data, either with one skill per item (A, top), or multiple skills per item (B, bottom).

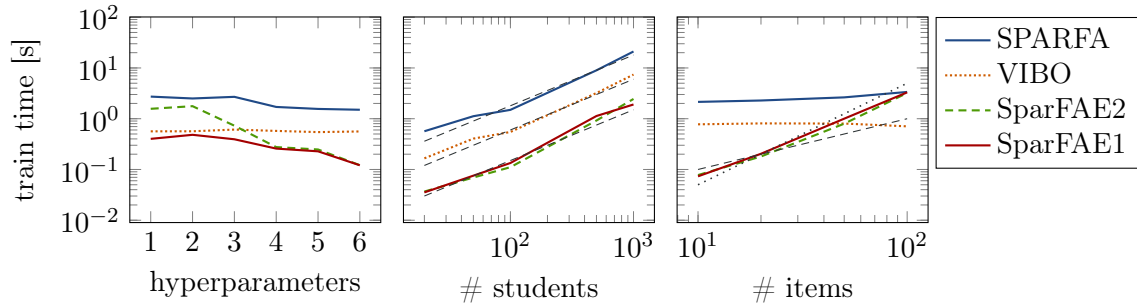


Figure 5: The effect of hyperparameter setting (left), number of students (center), and number of items (right) on training time for condition A. Gray dashed lines are linear references, the gray dotted line is a quadratic reference.

ber of items, the runtime for SPARFA and VIBO remains roughly constant, whereas it increases almost quadratically for both SparFAE variants. This is because the optimization for SPARFA and VIBO is dominated by iterations over students. By contrast, for SparFAE, every single gradient computation already depends linearly on n , and more items may also increase the number of required gradient computations until convergence, thus yielding the super-linear behavior.

4.2 NeurIPS 2020 education data

Next, we consider the NeurIPS 2020 education challenge data by Wang et al. [15]. The data set consists of multiple choice questions to assess mathematics knowledge. Items are grouped into different quizzes. We restricted the data set to the 948 items from task 3 of the challenge and quizzes with at least 50 students¹, which left 65 quizzes. On average, these quizzes contained 14.02 items and had 1675.06 students responding. To estimate the number of skills K , the first author analyzed all 948 items and assigned them to skills. This yielded 14 distinct skills, the most common ones being fractions (190 items), basic algebra (140 items), and algebra with variables (127 items). On average, quizzes involved 3.21 skills. For each quiz, we set K to the first-author estimate, but we upper-bounded K to be at most half the number of items in the quiz.

Based on the pre-defined \mathbf{Q} -matrix by the first author, we included two more baselines: A VIBO model, where we fixed the decoder matrix to the pre-defined \mathbf{Q} -matrix, and a SparFAE1 model, where we fixed the \mathbf{Q} -matrix and only trained the difficulty parameter for each item using logistic regression. We denote these methods as VIBO_f and SparFAE_f, respectively.

To perform hyperparameter optimization, we randomly set aside 10 quizzes and evaluated the AUC of all methods for all hyperparameter settings in Table 1 in a 10-fold cross-validation over students, that is, in each fold we used 90% of students as training data and 10% as test data. The hyperparameter settings which maximized AUC were 2 for SPARFA and SparFAE2, 3 for VIBO, and 4 for SparFAE1.

Next, we performed a 10-fold crossvalidation over students for the remaining 55 quizzes. Note that we can not evaluate r_b , r_θ , or $r_{\mathbf{Q}}$, because we have no access to a ground truth for b , θ , and \mathbf{Q} . However, we can evaluate the sparsity of \mathbf{Q} , that is, the fraction of zero entries. Sparsity is a rough proxy for the plausibility of a learned \mathbf{Q} -matrix because high sparsity indicates that \mathbf{Q} assigns items to distinct skills.

Table 2 reports the average performance measures across quizzes. Regarding AUC, Wilcoxon signed-rank tests revealed that SPARFA had the highest AUC, followed by SparFAE2, VIBO, SparFAE1, SparFAE_f, and finally VIBO_f ($p < 10^{-3}$ for all tests after Bonferroni correction). That being said, the AUC of all methods except SPARFA is very close (at most 2% difference between means). In terms of sparsity, SparFAE1 clearly outperforms SPARFA, VIBO, and SparFAE2 ($p < 10^{-3}$). Note that VIBO does not achieve any sparsity, as it does not encourage sparsity during train-

¹We also excluded one outlier quiz with more than 100 items and a lot of missing data.

ing. The sparsity of the pre-defined \mathbf{Q} -matrix was very high (94%) as it assigned each item to only one skill.

With respect to training time, SparFAE1 is considerably faster than SPARFA (ca. 15x), VIBO (ca. 4x), and SparFAE2 (ca. 8x). In terms of prediction time, SparFAE1, SparFAE2, and VIBO perform similarly as their prediction scheme is almost the same (although SparFAE1 is still significantly faster, $p < 10^{-3}$). Only SPARFA is much slower (ca. 300x) because it needs to fit new ability parameters to new students for each prediction.

Finally, we analyzed the relation of AUC to the numbers of students, items, and skills, as well as the amount of missing data in quizzes. Fig. 6 displays scatter plots, where each dot represents one quiz and lines show linear fits. Interestingly, the behavior is very similar for all methods. The linear correlation is $r \approx 0.3$ with number of students ($r = 0.4$ for VIBO; $p < 0.05$), $r \approx -0.4$ with number of items ($p < 0.01$), $r \approx 0.6$ with number of skills ($p < 10^{-3}$), and around zero with the amount of missing data (insignificant). This is in line with our results on synthetic data. The strong correlation with the number of skills is explained by the fact the methods have more parameters to fit the data when we increase K .

4.3 Math assessment data

In a final experiment, we evaluated the ability of SparFAE1 to identify a fitting \mathbf{Q} -matrix in comparison to an expert-designed \mathbf{Q} -matrix on real data. To that end, we used data from $m = 30$ students (ages 16-19) on a math assessment test for vocational education in chemistry². The test consisted of $n = 21$ questions, covering $K = 5$ topics, namely basic algebra, fractions, equation solving for a single variable, text tasks with two variables, and (linear) functions. Fig. 7 (top) shows the assignment of items (x-axis) to these five topics (y-axis) as provided by the test designers.

We applied a slightly adapted variant of SparFAE1 with the regularization $\sum_{k=1}^K \left(\sum_{j=1}^n q_{j,k} - 1 \right)^2$, that is, we punished deviations of the column sums from 1, thereby encouraging orthogonality in \mathbf{Q} . As regularization strength, we set 1. We performed 30 repeats of SparFAE1 and then selected the \mathbf{Q} -matrix which maximized accuracy in a leave-one-out crossvalidation over students (the resulting best accuracy was 89%).

The learned \mathbf{Q} -matrix is shown in Fig. 7 (bottom). We observe that the matrix assigns every item to only one skill, in line with the expert prediction. We further observe that—in line with the experts—the learned \mathbf{Q} tends to group items for the basic topics (basic algebra and fractions) together and tends to avoid grouping items for basic topics with items for advanced topics. However, there are also notable differences to the expert \mathbf{Q} -matrix. In particular, SparFAE1 merges basic algebra and fractions into one skill (except for item 8, which is in skill 4), and includes items 13 and 14. Overall, skill 1 accumulates relatively easy tasks without text- and function components. All other skills contains items which required text comprehension and/or un-

²https://projekte.provadis.de/showroom/provadis/Mathematik_Orientierungstest/online

Table 2: Performance measures on the NeurIPS 2020 education data.

method	AUC	sparsity	training time [s]	prediction time [ms]
VIBO _f	0.88 ± 0.05	0.94 ± 0.00	8.01 ± 5.59	1.31 ± 2.76
SparFAE _f	0.88 ± 0.04	0.94 ± 0.00	0.05 ± 0.03	0.15 ± 0.13
SPARFA	0.93 ± 0.05	0.16 ± 0.06	31.0 ± 20.9	633 ± 444
VIBO	0.89 ± 0.05	0.00 ± 0.00	7.83 ± 5.12	0.31 ± 0.18
SparFAE2	0.90 ± 0.05	0.33 ± 0.10	15.7 ± 15.9	0.20 ± 0.13
SparFAE1	0.89 ± 0.04	0.46 ± 0.13	1.94 ± 1.78	0.19 ± 0.12

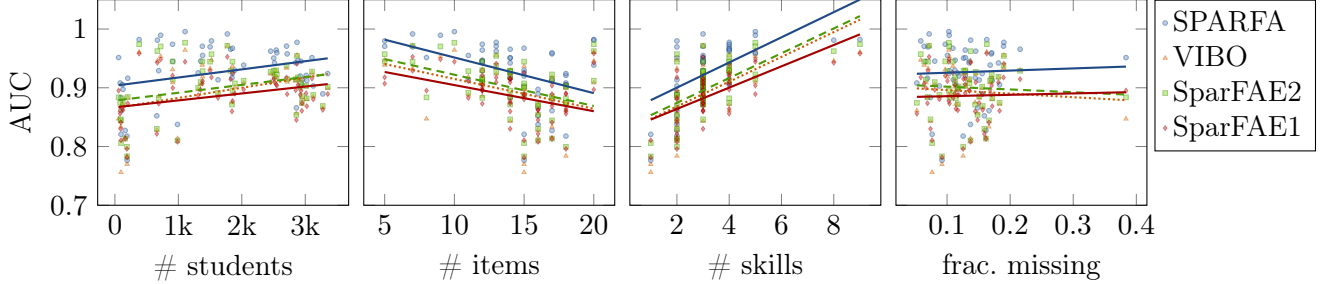


Figure 6: Scatter plots of AUC versus quiz metadata (from left to right: number of students, number of items, number of skills, and fraction of missing data). Lines indicate linear regression fits.

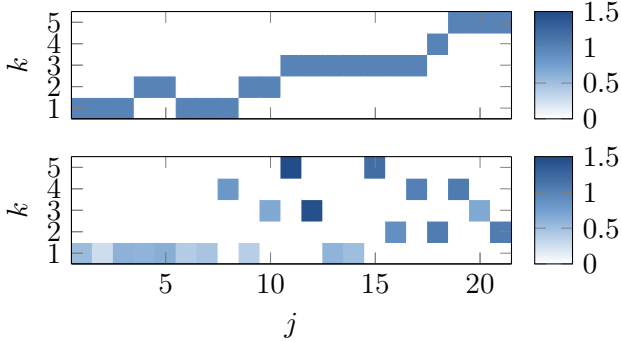


Figure 7: The expert-designed Q -matrix (top), and the learned Q -matrix via SparFAE1 (bottom) for the math assessment data.

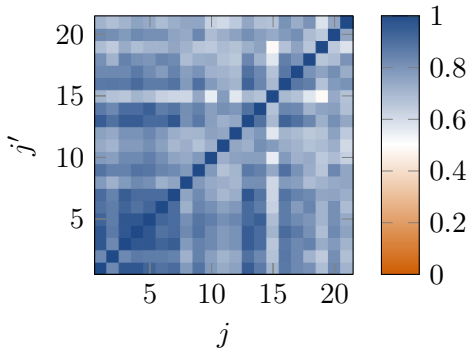


Figure 8: The item-to-item correlations for the math assessment data.

understanding of functions, but the correspondence to expert-defined skills is less obvious.

To gain deeper insight into the learned Q -matrix, we inspected the item-to-item correlations $c_{j,j'} = \frac{\sum_{i=1}^m x_{i,j} \cdot x_{i,j'}}{\sqrt{\left(\sum_{i=1}^m x_{i,j}\right) \cdot \left(\sum_{i=1}^m x_{i,j'}\right)}}$, which are shown in Fig. 8. We observe that items 1-7, 9, and 13-14 exhibit relatively high pairwise correlation, explaining why SparFAE1 grouped them together in skill 1.

Skill 2 groups items 16 and 18, which are both text problems covering variable solution problems, but it also includes item 21, which is a question on functions. Inspecting the correlation matrix, we observe that item 21 generally exhibits low correlation, except for items 16 and 18, which explains the grouping.

Skill 3 groups items without obvious mathematical connection. Item 10 is a fraction problem, item 12 is a variable algebra problem, and item 20 is a function problem. Further, these items exhibit only moderate pairwise correlation. However, the only items with higher correlations are already contained in skill 1 and are, thus, unavailable for skill 3, thus indirectly explaining the grouping.

Skill 4 contains a variable algebra item (8), an equation solving problem (17), and a function problem (19). General variable algebra capacity (8) plausibly enhances equation solving (17) but the function question (19) seems less connected. The correlation matrix reveals that item 19 has generally low correlations, except for items 3, 7, 14, and 17, explaining its grouping with item 17.

Skill 5 contains two equation problems, one symbolic (11) and one text-based (15). Further, item 15 has very low

correlations with any other item, except for items 9 and 11, and 20, which explains the grouping with item 11.

Overall, we observe that the learned \mathbf{Q} -matrix tended to group more basic items together and more advanced items together, in line with expert opinion. Sometimes, the learned \mathbf{Q} matrix groups items which do not have an obvious connection, content-wise. In such cases, we could explain the grouping by inspecting the item-to-item correlation matrix.

5. DISCUSSION AND CONCLUSION

We proposed a novel method for factor analysis which extends Sparse Factor Analysis (SPARFA) [7] to an autoencoder approach. Hence, we call our proposed method Sparse Factor Autoencoder (SparFAE). More specifically, our approach encodes student responses to abilities via a linear map \mathbf{A} and decodes it again to predicted responses via a multi-dimensional item response theory model with a linear skill-to-item map \mathbf{Q} . Like SPARFA, our approach encourages sparsity in the \mathbf{Q} -matrix via non-negativity constraints and L1 regularization. In contrast to SPARFA, we do not need to fit new ability parameters for new students. Instead, we can simply apply \mathbf{A} , which automatically yields the desired ability parameters. We investigated two versions of SparFAE: One with separate matrices \mathbf{A} and \mathbf{Q} for encoding and decoding (SparFAE2), and one where we set $\mathbf{A} = \mathbf{Q}^T$, that is, we use the \mathbf{Q} -matrix for both encoding and decoding (SparFAE1).

In experiments on synthetic as well as real data, we showed that SparFAE1 is considerably faster than SPARFA, variational autoencoding [16], and SparFAE2. SparFAE1 also achieves higher sparsity, and higher correlation with ground truth \mathbf{Q} -matrices and student abilities. This comes at the price of slightly lower AUC and less accuracy in recovering ground truth difficulties. We also observed that AUC differences between autoencoder variants were quite small, whereas SPARFA achieved noticeably higher AUC, indicating that student-specific ability parameters allow for a better fit of the data than autoencoding. We also compared the learned \mathbf{Q} -matrix via SparFAE1 with an expert \mathbf{Q} matrix on a math assessment test, revealing some overlap but also meaningful differences which could be explained by item-to-item correlations.

Overall, our results indicate that SparFAE1 is a promising method for fast factor analysis, especially when each item in a test only refers to a single skill. As such, we believe that it can be an interesting tool for test designers who wish to analyze the factor structure of their test on a sample of students. While the learned \mathbf{Q} -matrix should still be interpreted with care, it can uncover latent item relationships (as we saw on the math assessment data). Our results also motivate the use of \mathbf{Q} -matrices for both decoding and encoding, which can serve as a starting point for future research.

Limitations of SparFAE1 lie in the slightly lower AUC compared to other autoencoders, the ability to recover ground truth difficulty parameters, and the superlinear scaling with respect to the number of items. Future work could address each of these shortcomings. Further, our experimental evaluation is limited to multiple choice math assessment questions. Future work should include further data sets from

other educational domains to ensure that SparFAE1 generalizes. Finally, just as any autoencoders, SparFAE1 makes the assumption that abilities do not change during a test. Future work may consider more dynamic settings, e.g. by incorporating concepts from performance factor analysis or knowledge tracing models.

Acknowledgements

Funding by the German Ministry for Education and Research (BMBF) under grant number 21INVI1403 (project KIPerWeb) is gratefully acknowledged.

6. REFERENCES

- [1] F. Baker and S.-H. Kim. *Item Response Theory: Parameter Estimation Techniques*. CRC Press, Boca Raton, FL, USA, 2 edition, 2004.
- [2] T. Barnes. The \mathbf{Q} -matrix method: Mining student response data for knowledge. In *Proceedings of the AAAI 2005 Educational Data Mining Workshop*, pages 1–8, 2005.
- [3] G. Converse, M. Curi, and S. Oliveira. Autoencoders for educational assessment. In S. Isotani, E. Millán, A. Ogan, P. Hastings, B. McLaren, and R. Luckin, editors, *Proceedings of the International Conference on Artificial Intelligence in Education (AIED)*, pages 41–45, 2019.
- [4] S. Embretson and S. Reise. *Item response theory for psychologists*. Psychology Press, New York, NY, USA, 2000.
- [5] R. Hambleton and H. Swaminathan. *Item response theory: Principles and applications*. Springer Science+Business Media, New York, NY, USA, 1985.
- [6] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv*, 1312.6114, 2014.
- [7] A. S. Lan, A. E. Waters, C. Studer, and R. G. Baraniuk. Sparse factor analysis for learning and content analytics. *Journal of Machine Learning Research*, 15(57):1959–2008, 2014.
- [8] D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.
- [9] R. Liu, A. C. Huggins-Manley, and L. Bradshaw. The impact of \mathbf{Q} -matrix designs on diagnostic classification accuracy in the presence of attribute hierarchies. *Educational and Psychological Measurement*, 77(2):220–240, 2017.
- [10] R. P. McDonald. A basis for multidimensional item response theory. *Applied Psychological Measurement*, 24(2):99–114, 2000.
- [11] P. I. Pavlik, H. Cen, and K. R. Koedinger. Performance factors analysis – a new alternative to knowledge tracing. In *Proceedings of the International Conference on Artificial Intelligence in Education (AIED)*, page 531–538, 2009.
- [12] Y. Sun, S. Ye, S. Inoue, and Y. Sun. Alternating recursive method for \mathbf{Q} -matrix learning. In *Proceedings of the 7th International Conference on Educational Data Mining (EDM)*, pages 14–20, 2017.
- [13] K. K. Tatsuoka. Rule space: An approach for dealing with misconceptions based on item response theory. *Journal of Educational Measurement*, 20(4):345–354, 1983.

- [14] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, et al. Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature methods*, 17(3):261–272, 2020.
- [15] Z. Wang, A. Lamb, E. Saveliev, P. Cameron, Y. Zaykov, J. M. Hernández-Lobato, R. E. Turner, R. G. Baraniuk, C. Barton, S. P. Jones, S. Woodhead, and C. Zhang. Diagnostic questions: The NeurIPS 2020 education challenge. *arXiv*, 2007.12061, 2020.
- [16] M. Wu, R. L. Davis, B. W. Domingue, C. Piech, and N. Goodman. Variational item response theory: Fast, accurate, and expressive. In A. Rafferty, J. Whitehill, V. Cavalli-Sforza, and C. Romero, editors, *Proceedings of the 13th International Conference on Educational Data Mining (EDM)*, pages 257–268, 2020.
- [17] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.

Exploring Common Trends in Online Educational Experiments

Ethan Prihar
Worcester Polytechnic Institute
100 Institute Road
Worcester, Massachusetts
ebprihar@wpi.edu

Stacy Shaw
Worcester Polytechnic Institute
100 Institute Road
Worcester, Massachusetts
sshaw@wpi.edu

Manaal Syed
Worcester Polytechnic Institute
100 Institute Road
Worcester, Massachusetts
msyed@wpi.edu

Adam Sales
Worcester Polytechnic Institute
100 Institute Road
Worcester, Massachusetts
asales@wpi.edu

Korinn Ostrow
Worcester Polytechnic Institute
100 Institute Road
Worcester, Massachusetts
ksostrow@wpi.edu

Neil Heffernan
Worcester Polytechnic Institute
100 Institute Road
Worcester, Massachusetts
nth@wpi.edu

ABSTRACT

As online learning platforms become more ubiquitous throughout various curricula, there is a growing need to evaluate the effectiveness of these platforms and the different methods used to structure online education and tutoring. Towards this endeavor, some platforms have performed randomized controlled experiments to compare different user experiences, curriculum structures, and tutoring strategies in order to ensure the effectiveness of their platform and personalize the education of the students using it. These experiments are typically analyzed on an individual basis in order to reveal insights on a specific aspect of students' online educational experience. In this work, the data from 50,752 instances of 30,408 students participating in 50 different experiments conducted at scale within the online learning platform ASSISTments were aggregated and analyzed for consistent trends across experiments. By combining common experimental conditions and normalizing the dependent measures between experiments, this work has identified multiple statistically significant insights on the impact of various skill mastery requirements, strategies for personalization, and methods for tutoring in an online setting. This work can help direct further experimentation and inform the design and improvement of new and existing online learning platforms. The anonymized data compiled for this work are hosted by the Open Science Foundation and can be found at <https://osf.io/59shv/>.

Keywords

Randomized Controlled Experiments, Online Learning Platforms, Skill Mastery, Instructional Interventions, Online Tutoring

E. Prihar, M. Syed, K. Ostrow, S. Shaw, A. Sales, and N. Heffernan. Exploring common trends in online educational experiments. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 27–38, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license. <https://doi.org/10.5281/zenodo.6853041>

1. INTRODUCTION

The use of online learning platforms has increased rapidly in the past decade [37]. As online learning platforms grow to become a permanent fixture of educational systems, they have the potential to democratize education by providing high quality free or low-cost resources to compliment traditional classroom practices [1]. While in some cases online tutoring has been shown to be at least as effective as traditional in-person educational practices [33, 11, 16], there is still a need to validate the effectiveness of the various methods by which educational content is delivered to students. Placing an emphasis on objectively measuring the effectiveness of these emerging methods through randomized controlled experimentation is essential for ensuring that the quality of educational resources continues to increase.

This study works towards that endeavor by aggregating the results from 50,752 instances of 30,408 students participating in 50 different randomized controlled experiments conducted by various groups of researchers since February, 2019 within the online learning platform ASSISTments. In these experiments, K-12 students were randomized between different conditions as they completed online mathematics assignments. These conditions changed factors such as students' assignment completion requirements, the format of the tutoring students' received when struggling with the assigned problems, and the types of interactions students could have within their assignment. While these types of studies have been conducted in ASSISTments before [34, 40], this work goes beyond reporting the results of each individual study, and instead aggregates the results of these studies together, ultimately investigating 19 different research questions across 50 randomized controlled experiments. To achieve this, the following steps were taken.

1. Identify the independent measures of every condition in each experiment.
2. Normalize the dependent measures of all the experiments so they can be compared to one another.
3. Combine the data from different experiments when the research questions of the experiments match.

4. Determine the effects of the various experimental instructional interventions using these combined datasets.

The results of this aggregate analysis revealed actionable trends that can contribute to a broader understanding of the effectiveness of different educational interventions, help direct further experimentation, and inform the design and improvement of new and existing online learning platforms.

2. BACKGROUND

2.1 Educational Experiments

Experiments revolving around educational practices have been conducted since the late 19th century [49]. These early experiments, conducted by William James, Edward Thorndike, and Alfred Binet along with others, focused on determining individual differences between students, why they occur, and what methods teachers can employ to improve educational outcomes for them [49]. By the early 20th century, with the increased accessibility of formal education, educational experiments were more focused on improving teaching methods [49] and connecting cognitive psychology to classroom practices [24]. These studies investigated the differences in learning between students of varying socioeconomic levels [7], the effect of increasing student autonomy in the classroom [29], and the value of assessment in learning [21].

In the years following these studies, theories on educational development, classroom practice and structure, and how to approach individual differences between students were developed. In particular, research around effective feedback has proven to increase performance [23], interest in learning [9], as well as increasing students' abilities to self-regulate their learning [35]. These studies varied in the types of feedback students receive [9], level of specificity and frequency [48], level of praise present in the feedback [8], and what types of students benefit the most from certain types of feedback [14]. Data for these studies were collected from classroom observations of verbal feedback, collections of written feedback, and results on written assessments.

2.2 Experimentation within Online Learning Platforms

Computer-assisted instruction in education has been studied since the 1960s [47], results of these early studies show that providing specific, targeted feedback to student responses improves retention of information [19, 43]. In more recent years, educational data mining research has grown significantly, with large scale implementation of online A/B testing in web applications allowing thousands of users to be randomized into conditions simultaneously [3, 5]. With the rapid adoption of computers in the classroom in the past two decades, educational researchers now have access to an abundance of data on students. Online learning platforms track students' performance, demographics, interactions within the platform, statistics on content usage, feedback, and more [38]. Additionally, during the 2020-2021 school year many schools that had not previously used online learning platforms migrated to online learning platforms as a result of the COVID-19 pandemic [28]. This increase

in the size and scope of available data has made it possible to gain insights into educational practices that were not previously possible with traditional methods.

Recent studies have focused on predicting student outcomes, improving domain specific content, examining the effects of different kinds of pedagogical support, and advancing knowledge about how people learn [5]. Similar to early studies on computer-assisted learning, learning analytics research aims to determine what types of feedback and presentations work well for what types of students, in other words, discovering the potential for personalization in online learning platforms [31, 5]. Prior studies on personalization show the benefit of explanatory feedback over corrective feedback for novice students [31], differences in effect of feedback between male and female students [32], and the effects of immediate and delayed feedback for students with different prior knowledge levels [45, 10]. Additionally, by taking advantage of recent advances in data collection, research has been able to focus on determining methods for personalizing based on students characteristics, such as district locale and student interaction data [2] and what types of crowdsourced content is effective for students [39]. This work provides another data-intensive analysis on the effectiveness of different aspects of online learning platforms, but unlike the aforementioned analyses, this work compiled data from dozens of studies performed within an online learning platform instead of focusing on a single study. This revealed trends across experiments that provided deeper insight into the effectiveness of various instructional interventions and online tutor designs.

2.3 ASSISTments and E-TRIALS

The data in this work comes from ASSISTments, an online learning platform that focuses on providing teachers with mathematics content and resources to effectively manage their students. Within ASSISTments, teachers have the option to assign problem sets and skill builders to their students. Problem sets are a series of mathematics problems that must all be completed, in order, to finish the assignment. These problem sets come from various open educational resources for mathematics such as Engage New York, Illustrative Mathematics, and The Utah Middle School Math Project. Skill builders are assignments that focus on a specific mathematics skill. When students complete skill builders they are given a series of problems on the same mathematics skill until they get a specific number of problems correct in a row. Usually students must answer three problems correct in a row to finish the assignment, but this number is configurable by the teacher.

Regardless of whether the student is assigned a problem set or skill builder, they complete their assignment in the ASSISTments tutor [20]. In the tutor, students receive immediate feedback when they submit a response to a problem, which informs them if they are correct [27]. In addition to this immediate feedback, students are able to request tutoring, which is available to them at any point during their completion of a problem regardless of whether or not they have already attempted the problem. Tutoring comes in the form of hints, which are a series of messages the student can request, one at a time, that explain how to solve parts of the problem; explanations, which are full worked solutions to the problem; examples, which are full worked solutions

of a similar problem; common wrong answer feedback messages, which explain how to correct a specific error made by the student; and scaffolding, which breaks the problem down into a series of simpler problems that guide the student through how to solve the original problem [27]. These different types of tutoring strategies can come in the form of videos, images, or text. An example of a student receiving a text-based explanation within the ASSISTments tutor is shown in Figure 1. Once students have finished their assignment, teachers are provided with reports that aggregate information such as how each student progressed through the assignment and what the class’ most common mistakes were.

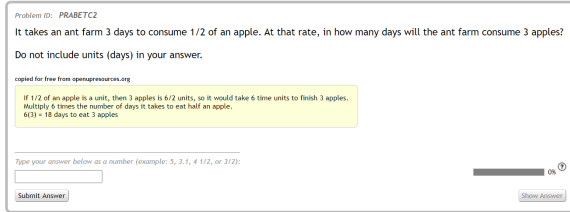


Figure 1: A student’s view of the ASSISTments tutor after requesting tutoring and receiving a text-based explanation.

The variety of assignments and tutoring strategies that can be delivered to students through ASSISTments provides opportunities to explore various research questions in learning science, educational psychology, and human-computer interaction. A research test-bed, E-TRIALS (an EdTech Research Infrastructure to Advance Learning Science), was built to deploy randomized controlled experiments in classwork and homework settings at scale within ASSISTments [25]. Since 2005, researchers have been able to create and modify problem sets, skill builders, and tutoring strategies. The modified content contains the original content within it, but adds experimental conditions. For example, a researcher could modify a skill builder for calculating the area of a triangle to randomly provide students with text-based or video-based hints. Teachers assign the modified content as if it were the original, and when teachers assign these modified assignments, students will be randomized (on an individual basis, not at the class level) to one of multiple conditions. This allows researchers to evaluate the impact of different pedagogical decisions on students’ learning [41]. The experiments run in ASSISTments cover a wide scope of research questions that range from whether offering students a choice in the difficulty of their instruction improves learning, to whether providing students with worked examples of similar problems is more effective than direct advice on the problem they are struggling with, to whether changing the number of problems students are required to complete affects their learning [44]. The following analysis of E-TRIALS experiments provides insight into the current state of experimentation within online learning platforms and can help inform the design of future experiments.

3. EXPERIMENT DATASET

The dataset used in this work comes from 50,752 instances of 30,408 unique students who participated in 50 E-TRIALS experiments since February, 2019. In addition to recording the purpose of the experiment, the experimental con-

dition each student was placed in, and the resulting dependent measure, the dataset also includes information on students’ performance within ASSISTments prior to participating in the experiment, the prior performance of the students’ classes, the experience of their teachers, and an indicator of their socioeconomic status. Socioeconomic status is indicated by a student’s school district’s Opportunity Zone status, which is a particular tax classification in the United States of America that indicates whether a region has opportunities for economic growth. The regions in opportunity zones are typically low-income regions with fewer educational resources [15]. The Opportunity Zone status for each student was determined using the domain name of their teacher’s school-provided email address. No demographic information was requested from students using ASSISTments to preserve their anonymity and prevent any bias associated with answering questions on how they identify themselves. The full set of features collected for each participant is shown in Table 1. In addition to containing features for each experiment participant, the dataset contains information on the independent and dependent measures used in the various experiments, which had to be aggregated in order to determine the common trends among the 50 experiments. The details of these independent and dependant measures and how they were aggregated are discussed in Sections 4.1 and 4.2.

4. METHODOLOGY

Due to the diversity in research questions, independent and dependent measures, and structure of the experiments, the first step to evaluate their overall trends was to identify similar conditions within multiple experiments. This process involved documenting each condition of each experiment and identifying when different experiments had an identical pair of conditions or the same research question. The second step was to normalize the various dependent measures such that they all represented similar metrics and used the same scale.

4.1 Pooling Experiment Data

To pool experimental data together, similar experiments had to be identified. To do this, every condition from every experiment was documented such that data from multiple experiments that each had an identical pair of conditions or research question could be aggregated. For example, if the following three experiments were run in ASSISTments, then Experiment 1 would have six documented conditions (one condition for each of the hint types for both choice and no choice), Experiment 2 would have two conditions (one for text-based hints and one for video-based hints), and Experiment 3 would have four conditions (one condition for each text color for both choice and no choice).

- **Experiment 1:** Randomize between A: giving students a choice of no hints, text-based hints, or video-based hints, or B: randomly selecting which type of hint to give them.
- **Experiment 2:** Randomize between A: text-based hints, or B: video-based hints.
- **Experiment 3:** Randomize between A: giving students a choice of black or red text color, or B: randomly selecting the text color.

Table 1: The Features Calculated for each Instance of a Student Participating in an Experiment

Feature Name	Description
Experiment Condition	An indication of which condition students are in.
Student Prior Started Skill Builder Count	Number of skill builders previously started by students.
Student Prior Skill Builder Percent Completed	Percent of skill builders completed by students.
Student Prior Started Problem Set Count	Number of problem sets previously started by students.
Student Prior Problem Set Percent Completed	Percent of problem sets completed by students.
Student Prior Completed Problem Count	Total number of problems completed by students.
Student Prior Median First Response Time	Students' median time to submit an answer to a problem.
Student Prior Median Time On Task	Students' median time to complete a problem.
Student Prior Average Attempt Count	Student's average attempts required to complete a problem.
Student Prior Average Correctness	The fraction of problems students answered correctly.
Class Age In Days	The number of days classes existed in ASSISTments.
Class Student Count	The number of students in the class.
Class Prior Started Skill Builder Count	Number of skill builders previously started by classes.
Class Prior Skill Builder Percent Completed	Percent of skill builders started by classes that were completed.
Class Prior Started Problem Set Count	Number of problem sets previously started by classes.
Class Prior Problem Set Percent Completed	Percent of problem sets started by classes that were completed.
Class Prior Completed Problem Count	Total number of problems completed by classes.
Class Prior Median First Response Time	Class' median time to to submit an answer to a problem.
Class Prior Median Time On Task	Class' median time to complete a problem.
Class Prior Average Attempt Count	Class' average attempts required to complete a problem.
Class Prior Average Correctness	The fraction of problems classes answered correctly.
Teacher Account Age In Days	The number of days teachers have had an ASSISTments account.
Experiment Id	The experiment students participated in.
Opportunity Zone	The school district's Opportunity Zone status [15].

In addition to documenting the conditions for the three experiments, Experiments 1 and 3 would be recorded as having the higher-level research question “Choice vs. No Choice” and Experiment 2 would be recorded as having no higher-level research question. To combine the results of these three experiments, students randomized to the text-based hint option of Condition B of Experiment 1 would be combined with students randomized to Condition A of Experiment 2 and students randomized to the video-based hint option of Condition B of Experiment 1 would be combined with students randomized to Condition B of Experiment 2. These groups would be used to evaluate the overall effect of giving video-based hints compared to text-based hints. Additionally, students randomized to Condition A of Experiments 1 and 3 would be grouped, and students randomized to Condition B of Experiments 1 and 3 would be grouped. These two groups would be used to evaluate the overall effect of offering students a choice.

When performing this aggregation on the real experiments, many experiments were too unique to have similar experimental conditions as other experiments. Additionally, some experiments were created incorrectly in ASSISTments or had broken links to videos, leading students to never be randomized to a condition. Even though 103 experiments have been deployed in ASSISTments since 2019, only 50 had at least one condition similar to a condition in another experiment and were complete enough to be included in the analyses. After parsing through the data and removing poorly structured and broken experiments, the most common research questions were selected for further analysis. Table 2 shows the selected research questions and statistics on the data aggregated to evaluate the research questions. Students were typically divided evenly between the different

conditions, but for the research question “Emotion vs. No Emotion”, there were six conditions that included positive emotional content and two conditions that did not include emotional content, which is why about three fourths of students are placed in the treatment condition.

The six research questions containing the phrase “Correct for Mastery” all investigated differences in the requirements to complete a skill builder assignment. In a skill builder, students must correctly answer a specific number of problems in a row to complete the assignment. The different values in these research questions represent the different number of problems students had to get correct in a row before finishing the assignment or completing a posttest. The six research questions that compare something to “Answer Only” investigated how six different tutoring strategies improved student learning compared to just giving struggling students the answer. Table 3 describes each tutoring strategy investigated by these research questions. The other seven research questions are not related to other research questions, but examined different aspects of the structure of assignments and tutoring in online learning platforms.

- **Video vs. Text** investigated the difference between providing two different types of tutoring which were almost identical, except in one condition the tutoring content was text-based, and in the other condition the same tutoring was provided in a video format.
- **Common Wrong Answer Feedback vs. No Feedback** investigated the effect of providing students with specific feedback messages when they submitted a common wrong answer to any of the the problems in their assignment.

Table 2: Research Questions Selected for Analysis

Research Question	Experiment #	Student #	% in Treatment
2 Correct for Mastery vs. 3 Correct for Mastery	4	1192	0.487
2 Correct for Mastery vs. 4 Correct for Mastery	4	1165	0.475
2 Correct for Mastery vs. 5 Correct for Mastery	3	846	0.483
3 Correct for Mastery vs. 4 Correct for Mastery	5	2030	0.492
3 Correct for Mastery vs. 5 Correct for Mastery	4	1683	0.494
4 Correct for Mastery vs. 5 Correct for Mastery	4	1681	0.495
Example vs. Answer Only	3	765	0.467
Explanation vs. Answer Only	1	85	0.471
Hint vs. Answer Only	5	1192	0.513
Scaffolding vs. Answer Only	7	2010	0.546
Video Example vs. Answer Only	1	366	0.484
Video Scaffolding vs. Answer Only	3	1033	0.509
Video vs. Text	5	2492	0.497
Common Wrong Answer Feedback vs. No Feedback	2	7046	0.497
Adaptive vs. Non-Adaptive	9	7754	0.498
Fill-In vs. Multiple Choice	2	4057	0.493
Choice vs. No Choice	9	12789	0.499
Emotion vs. No Emotion	2	1211	0.766
Motivational vs. Non-Motivational	14	12243	0.581

- **Adaptive vs. Non-Adaptive** investigated the impact of changing the difficulty of problems based on how well students performed at the beginning of their assignment. Students that got problems correct at the beginning were given more challenging problems than the students that got the beginning problems incorrect.
- **Fill-In vs. Multiple Choice** investigated the impact of requiring students to write the correct answer in themselves compared to selecting from multiple preset options when answering questions.
- **Choice vs. No Choice** investigated the impact of allowing students to choose which version of various configurations for their assignments they would complete.
- **Emotion vs. No Emotion** investigated the impact of including positive emotional phrases and images in the body of the problems in the assignment. For example, an emotional problem would say “Susan excitedly purchased three apples.” instead of “Susan purchased three apples.”.
- **Motivational vs. Non-Motivational** investigated the impact of interjecting motivational messages and videos into the assignment.

4.2 Normalizing Student Learning

In addition to identifying similar conditions and research questions, the different experiments dependent measures had to be normalized such that the results from one experiment could be compared to another experiment. Normally, it would be very difficult to combine dependent measures from different experiments, but conveniently, all of the experiment in ASSISTments are attempting to increase student learning, and therefore the various dependent measures are just different ways of measuring student learning and can thus be normalized and combined. In the various E-TRIALS experiments, there are five different dependent measures used, described in Table 4.

While all of these measures represent student learning, they do not all increase as student learning increases, nor do they all have the same range, nor do they all take into account when a student fails to complete the experimental assignment, which presumably means they learned the least. To rectify these concerns, Table 5 shows the function $f(x)$ applied to each of the dependent measures. After $f(x)$ is applied to the dependent measures, the values are z -scored within each experiment using the pooled standard deviation grouped by experimental condition. This ensured that all of the different measures of learning increased as student learning increased, had the same scale, and accounted for incomplete assignments. These transformations converted all the dependent measures into a measurement of how many standard deviations above or below average each student performed compared to other students that participated in the same experiment. $f(x)$ for problems to mastery is particularly complicated because unlike the other dependent measures, problems to mastery goes down the more a student learns, and problems to mastery is bounded in the range $[3, \infty)$. Therefore, to ensure that $f(x)$ for problems to mastery increases the more a student learns, problems to mastery was transformed by inverting it, then multiplying it by 3. However, this transformation alters problems to mastery non-linearly, so to correct some of the non-linearity, the square root is taken, which makes $f(x)$ appear linear in the range $[3, 10]$ where most of the results lie.

4.3 Evaluating Differences in Student Learning

To measure the effects of the various experimental treatment conditions, Cohen’s d [12] was used to calculate the effect size between the control and treatment conditions for each research question. To test for a difference between treatment and control, we ran ordinary least squares models and examined the associated p -values and 95% confidence intervals of the mean differences between conditions, and used Cohen’s d to capture the magnitude of any effect. This model was used to predict normalized student learning

Table 3: Descriptions of Different Tutoring Strategies

Tutoring Strategy	Description
Example	An explained solution to similar problem.
Explanation	An explained solution to the current problem.
Hint	Step-by-step advice on how to solve the current problem.
Scaffolding	A series of problems that break the current problem into smaller steps with explanations.
Video Example	An example recorded in a video instead of text.
Video Scaffolding	A scaffolding with explanations recorded in videos instead of text.

Table 4: Descriptions and Frequencies of the Dependent Measures used to Evaluate Student Learning

Dependent Measure	Frequency of Use	Description
Problems to Mastery	44%	# of problems the student completed to get n correct in a row.
Posttest Score	44%	% correct on posttest.
Learning Gains	7%	% correct on posttest - % correct on pretest.
Assignment Correctness	3%	# of problems correct / # of problems in condition.
Assignment Completion	1%	Binary indicator for if the student completed the assignment.

Table 5: Functions used to Scale the Dependent Measures Before z -Scoring

Dependent Measure	$f(x)$
Problems to Mastery	0 if incomplete else $\sqrt{\frac{3}{x}}$
Posttest Score	0 if incomplete else x
Learning Gains	0 if incomplete else $x + 1$
Assignment Correctness	x
Assignment Completion	x

based on the experiment condition the student was placed in, the experiment the student participated in, and features of the student, their class, their teacher, and their school district. Including fixed effects for which experiment the student participated in allowed the model to associate differences in normalized student learning between experiments with those coefficients, and not the experiment condition coefficient, helping to reduce noise from covariates. The inputs related to students, classes, teachers, and school districts also helped to remove noise from the experiment condition coefficient. For example, students with high prior knowledge performed better on the experimental assignments than students with low knowledge, and by including students’ prior knowledge in the model, the variability in students’ success based on their prior knowledge will be associated with the prior knowledge coefficient, and have a lesser effect on the treatment coefficient. Table 1 contains a full list of the features used to model the effects of the various experimental conditions. The “Experiment Condition” feature was used to determine the 95% confidence interval and p -value of the impact of the various experimental instructional interventions. When some features were not available, such as when students that had not previously used ASSISTments participated in the experiments, the missing values were filled using the average value across the data used to fit the model. This limited the extent to which the missing values biased the model’s coefficients.

5. RESULTS

5.1 Different Completion Requirements

Investigating the impact of different mastery requirements for skill builders found that requiring fewer problems led to higher student learning than requiring more problems, but that this effect is mostly due to students not completing the assignment when they were required to answer more problems correct in a row to proceed. Figure 2 shows the effect size and, in parentheses, the p -value of the effect of requiring students get different numbers of problems correct in a row. For example, the cell at row two, column three contains the effect size and p -value of requiring students get two problems correct in a row instead of three problems correct in a row. Figure 2 only shows significant positive effects when requiring students to complete two problems in a row correctly instead of three, four, or five.

The Effect of Changing Problem Completion Requirements on Normalized Student Learning (Column vs. Row)

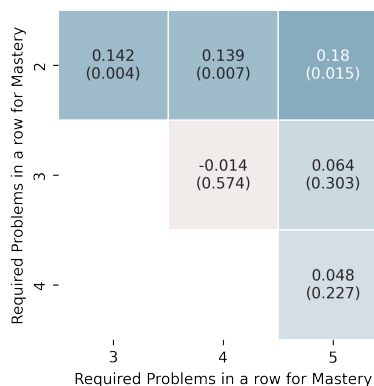
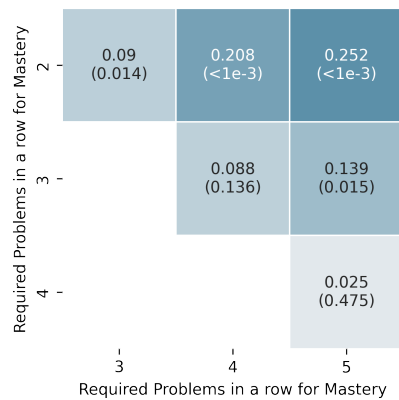


Figure 2: The effect of changing problem completion requirements on normalized student learning. Each cell contains the effect size, determined using Cohen’s d , and in parentheses, the p -value.

To investigate further, the effect of changing problem completion requirements on assignment completion and the effect of changing problem completion requirements on student learning for only students that completed the assignment were calculated. Figure 3 shows the results of these

analyses. Based on these results, there is no statistically significant effects on student learning for students that completed their assignment, regardless of how many problems they had to complete correctly in a row before finishing the assignment. The vast majority of the effects seen in Figure 2 come from more students failing to complete their assignment when having to complete more problems correct in a row. Essentially, when students have to complete more problems they are less likely to complete their assignment, but if students complete their assignment their learning will be unaffected by how many problems they had to complete.

The Effect of Changing Problem Completion Requirements on Assignment Completion (Column vs. Row)



The Effect of Changing Problem Completion Requirements on Normalized Student Learning for Students that Completed the Assignment (Column vs. Row)

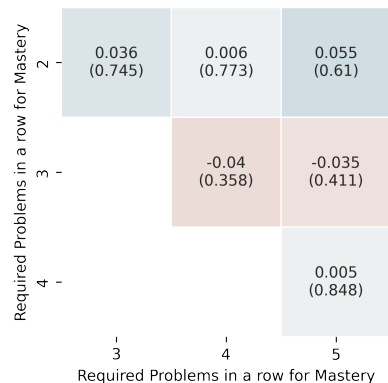


Figure 3: The effect of changing problem completion requirements on assignment completion (top) and normalized student learning for only students that completed the assignment (bottom). Each cell contains the effect size, determined using Cohen’s d , and in parentheses, the p -value.

One would expect that if any of these mastery requirements were a meaningful metric for determining if students had learned the material, then there would be a statistically significant difference in students’ learning between students that had to complete above or below a certain number of problems correct in a row. However, this was not the case. These results imply that a more sophisticated method could be necessary to evaluate whether students have mastered

the mathematics concepts present in their assignments. It may therefore be advisable to integrate Knowledge Tracing [13] or Performance Factors Analysis [36], which are both effective methods for evaluating students’ mastery of individual skills, into ASSISTments and other online learning platforms.

5.2 Different Tutoring Strategies

Investigating the effects of different types of tutoring on student learning found that most tutoring is effective, and that giving students tutoring instead of showing them the answer is more effective for low knowledge students than high knowledge students. Figure 4 shows the confidence interval, effect size, number of students, and p -value for the effect of giving students each type of tutoring instead of just providing the answer. The only tutoring strategy that had no significant impact on student learning was explanations, which had a wide confidence interval and relatively few participants.

Prior studies done in ASSISTments reported that lower knowledge students benefited more from scaffolding while higher knowledge students benefited more from short explanations [42]. Therefore, in addition to evaluating the effect of each of the above tutoring strategies on all students that participated in the experiments, the data from the experiments were divided into below and above average prior knowledge groups based on whether students’ prior average correctness was above or below the average of all students’ prior average correctness. Figure 5 shows the difference in the effectiveness of four of the six tutoring strategies for each of these groups of students. Only four of the six tutoring strategies from Figure 4 are included in these plots because the other two tutoring strategies were used in experiments that did not have any participants that had used ASSISTments previously, and therefore no prior average correctness was available for those students. The below average prior knowledge students consistently had statistically significant positive effects from being provided with tutoring and greater effect sizes for three out of the four tutoring strategies. These results agree with previous studies on the effectiveness of different tutoring strategies on different groups of students [42]. Additionally, Figure 5 shows that examples had the largest difference in their effectiveness between below and above average prior knowledge students and were the only tutoring strategy that had a statistically significant positive effect for below average prior knowledge students, but not for above average prior knowledge students.

Disparities in education, particularly in math, are often due to unequal access to opportunities to learn from highly qualified educators, otherwise known as the “opportunity gap” [17]. Although online learning platforms cannot replace a highly qualified educator, these results indicate that some online tutoring strategies can support in closing this opportunity gap for the most vulnerable students instead of just helping the more knowledgeable students succeed.

5.3 Other Instructional Interventions

Sections 5.1 and 5.2 covered two groups of related research questions, but there were many other research questions that did not fall into a group. Figure 6 shows the confidence intervals, number of participants, p -values, and effect sizes of these research questions. Of the various experiments, the ef-

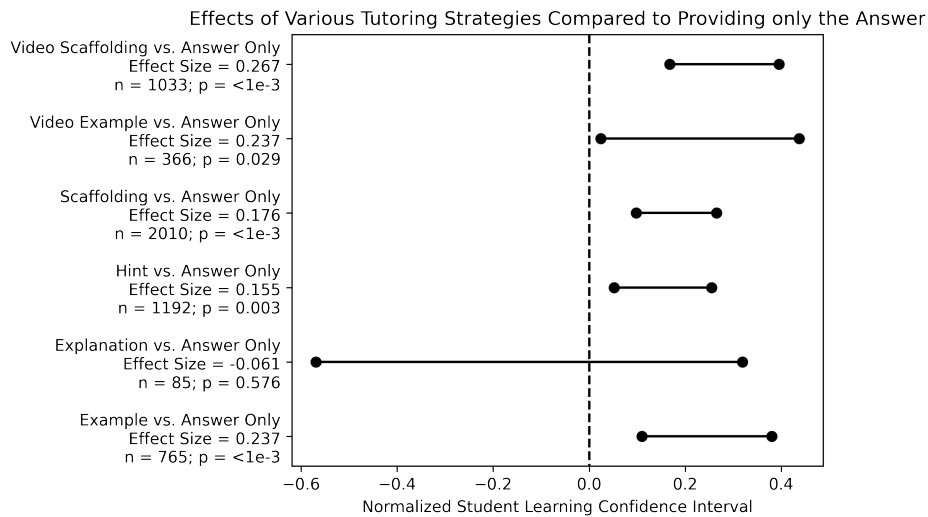


Figure 4: The effects of various tutoring strategies compared to providing only the answer. Effect size was determined using Cohen’s d , the confidence interval and p -value come from the experiment condition model coefficient.

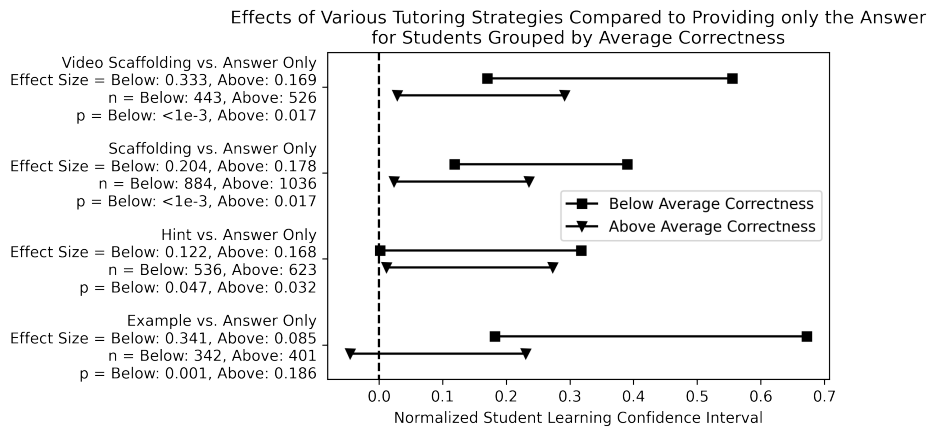


Figure 5: The effects of various tutoring strategies for below average and above average students. Effect size was determined using Cohen’s d , the confidence interval and p -value come from the experiment condition model coefficient.

fect of giving video-based tutoring compared to text-based tutoring had the largest effect size, with students learning more from video-based tutoring than text-based tutoring. This could have been due to videos being more engaging and not requiring students to also be proficient readers. Additionally, giving students mathematics problems with open responses, where they are not given optional answers to choose from, resulted in more learning than when they were given problems with multiple choices. This could have been due to the added difficulty of attempting to answer a problem without knowing what the potential solutions are. Another significant finding was that adapting students’ assignments based on their prior knowledge by altering the material given to them had a statistically significant positive effect, lending support to the idea that learning platforms should personalize students’ learning based on their prior knowledge, which has been found to be true in various studies and meta-analyses [42, 26]. Lastly, it was found that motivational messages have a negative impact on stu-

dents’ learning. This could be a result of students finding the messages distracting. However, students’ perceptions of the messages were not recorded as part of these experiments, and follow-up experiments should be performed to investigate further.

5.3.1 Video vs. Text

Although providing students with video-based tutoring instead of text-based tutoring resulted in an overall positive effect for all types of tutoring, it is possible this was due to a particularly large impact of receiving video instead of text for one type of tutoring strategy. Figure 7 shows the effect of providing video-based tutoring instead of text-based tutoring for the three types of tutoring strategies that were used in experiments where a video-based and text-based version of the same content was provided to students. Video-based scaffolding had the only significant positive effect on learning compared to a text-based control. Hints and examples had no statistically significant difference in their effective-

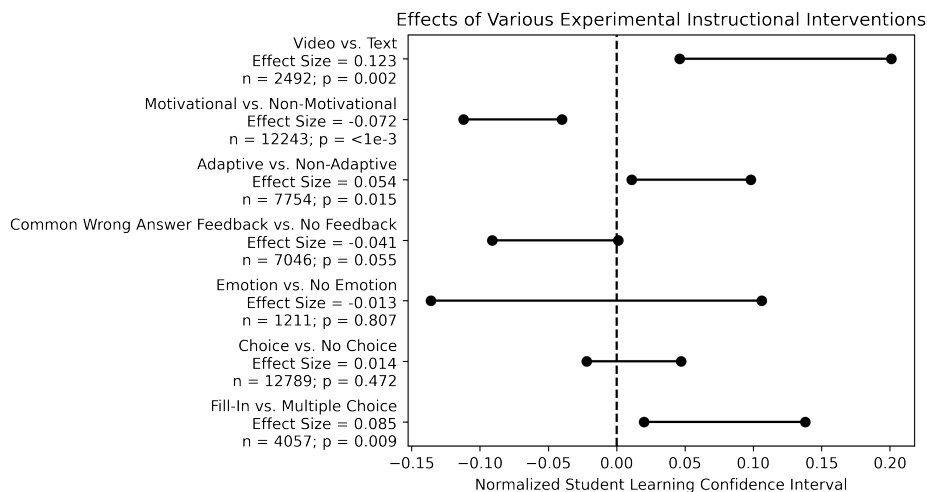


Figure 6: The effects of various experimental instructional interventions. Effect size was determined using Cohen’s d , the confidence interval and p -value come from the experiment condition model coefficient.

ness when video-based or text-based. From these results, one can infer that students benefit differently from different types of tutoring being video-based. Scaffolding offers a series of simpler problems to help students understand the problem they are struggling with. It could be that students are more likely to engage with videos that give them necessary context. The scaffolding videos ask students questions that they must solve to move on, without watching the videos, they cannot know what the question is. Students may not be as willing to watch videos that provide relevant, but not entirely necessary information on a problem they must solve.

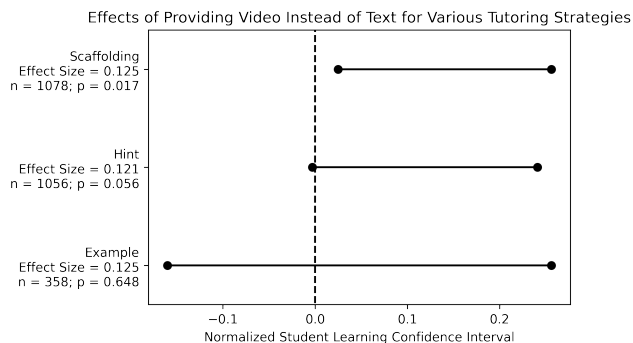


Figure 7: The effects of providing video instead of text for various tutoring strategies. Effect size was determined using Cohen’s d , the confidence interval and p -value come from the experiment condition model coefficient.

6. LIMITATIONS

The results in this work help to reinforce a foundation of knowledge on educational experimentation and can be used to influence the next generation of experimentation, but there are two notable limitations to the extent to which these results can be applied in the future. Firstly, the scope of the experiments analyzed in this work is limited to experiments conducted within ASSISTments. It could be that the user

interface of ASSISTments effects how beneficial certain instructional interventions are. For example, the way ASSISTments takes away partial credit for some tutoring strategies but full credit for others could impact the generalizability of these findings into a context where there is no scoring of student responses. All of the experiments also take place within skill builder assignments, in which students are given a series of similar problems on the same mathematics topic. The instructional interventions in the experiments analyzed in this work could have different effects on students completing assignments on topics outside of mathematics, or even a variety of mathematics topics within the same assignment. There could also be an issue generalizing these findings to contexts outside of online learning platforms. The differences between different tutoring strategies could be inconsequential if there is a teacher in the room to answer questions, and while the results of these experiments implied that motivational messages had a negative impact on learning, this was likely due to the distracting and impersonal nature of the motivational messages. Previous studies have shown the need for trust between teachers and their students and how this can lead to more motivated and academically successful students [4], but the trusting relationship needed for that impact is unlikely to exist between a student and a website.

Secondly, this work investigates many different research questions using data from a combination of experiments with similar, but not identical designs, which has increased the potential of discovering false positives in the analysis. This should influence the confidence that one has in the results of this work. While the results with effect sizes greater than 0.1 and p -values in range of 10^{-3} can likely be trusted, there are many weaker findings that some might consider significant while others may be more critical. By providing the sample sizes, effect sizes, confidence intervals, and p -values for every comparison carried out, for all the research questions investigated in this work, others can make an informed decision on the extent to which they should believe each of these findings, and which findings merit follow-up investigations and repeat experiments.

7. CONCLUSION

In this work, data from 50,752 instances of one of 30,408 students participating in one of 50 different experiments on a variety of instructional interventions conducted within ASSISTments were combined to investigate their impact on learning. Using this data, 19 different research questions regarding the effectiveness of these various instructional interventions were investigated, and this investigation revealed multiple actionable findings that can be used to design more effective online learning experiences.

The first insight discovered was that changing the number of problems students must get correct in a row to be considered as having mastered a skill had no impact on the learning gains of the students that were able to complete the assignment, but the more problems required, the more likely students were to stop doing the assignment before mastering the material, which overall decreased their learning gains. Based on this result, when creating mastery-based content, it might be better to use something like Knowledge Tracing [13] to evaluate mastery instead of forcing students to complete a fixed number of problems.

It was also discovered that across multiple experiments, the tutoring provided to students by ASSISTments had almost entirely a positive effect on students' learning compared to just giving students the answer when they were struggling. This falls in line with the larger findings from cognitive psychology that show students learn more when they productively struggle with solving problems, rather than being provided solutions [6]. Additionally, below average prior knowledge students benefited more from this tutoring overall than above average prior knowledge students, which can help to close opportunity gaps, and for all students, when scaffolding problems were video-based, they had a larger positive impact than when they were text-based. These results could help inform developing platforms on how to allocate limited resources when creating tutoring. For example, creating new tutoring could be prioritized for remedial courses, and the extra effort of making video-based tutoring could be saved for scaffolding.

Another insight from these analyses was that students showed greater learning patterns when they completed open-response questions rather than multiple choice questions. This corroborates some research that finds that memory and learning benefit most from free recall of information (e.g. answering an open-ended question) compared to cued-recall (e.g. multiple-choice items) during learning [22, 30]. Based on this, online learning platforms could move away from multiple choice questions when possible.

This study also found that adjusting students' assignments based on their prior knowledge level had a positive effect on their learning. This supports the idea that personalized learning can help students. Within ASSISTments, a previous study found that high-knowledge students learned more from explanations, while low-knowledge students learned more from scaffolding [42]. This is one example of how personalization based on prior knowledge within online learning platforms has been found to be effective in the past. Additionally, a meta-analysis of studies that measured the learning gains of students after grouping them by ability level found

that the instructional material was more than twice as effective when it was tailored to the students' ability levels than when it was held constant for all students [26]. The results of this study agree with prior work, and imply that personalizing students' education based on their prior knowledge increases their learning.

Another interesting result from these experiments was that motivational messages had a negative impact on learning. Past research has found positive effects of motivational interventions for some students, so why might these studies show a negative effect? One speculation is that the motivational videos may have unintentionally produced an effect similar to what is referred to as "seductive details" or highly engaging but unrelated information that is unnecessary for learning [46]. Including seductive details can lead to worse performance both in the classroom and in online learning environments [18], and is theorized to disrupt learning by redirecting attention away from the material and toward superfluous information, stopping students from appropriately allocating cognitive resources to the educational material. Providing motivational videos in the middle of the learning period may have produced a negative effect on learning because it disrupted cognitive processes necessary for learning, but more research is needed to fully investigate this and other possible mechanisms at play.

In addition to these results' capacity for improving online learning platforms, these results can help inform the next round of experimentation within online learning platforms. Future experiments could continue to investigate the inconclusive findings in this analysis, and expand upon the conclusive findings. For example, more types of problems besides multiple choice and open response problems could be compared to each other, and the effectiveness of different tutoring strategies could be investigated for differences based on subject matter or grade level. Through these analyses, learning platforms can continue to improve their design and increase their positive impact for all students that use them.

8. ACKNOWLEDGMENTS

We would like to thank NSF (e.g., 2118725, 2118904, 19506-83, 1917808, 1931523, 1940236, 1917713, 1903304, 1822830, 1759229, 1724889, 1636782, & 1535428), IES (e.g., R305N21-0049, R305D210031, R305A170137, R305A170243, R305A1-80401, & R305A120125), GAANN (e.g., P200A180088 & P200A150306), EIR (U411B190024 & S411B210024), ONR (N00014-18-1-2768) and Schmidt Futures. None of the opinions expressed here are that of the funders.

9. REFERENCES

- [1] D. Acemoglu, D. Laibson, and J. A. List. Equalizing superstars: The internet and the democratization of education. *American Economic Review*, 104(5):523–27, 2014.
- [2] S. Adjei, K. Ostrow, E. Erickson, and N. Heffernan. Clustering students in assistments: exploring system-and school-level traits to advance personalization. In *The 10th International Conference on Educational Data Mining*, pages 340–341. ERIC, 2017.
- [3] R. S. Baker, K. Yacef, et al. The state of educational data mining in 2009: A review and future visions.

- Journal of educational data mining*, 1(1):3–17, 2009.
- [4] A. Bennett, B. L. Bridglall, A. M. Cauce, H. T. Everson, E. W. Gordon, C. D. Lee, R. Mendoza-Denton, J. S. Renzulli, and J. K. Stewart. All students reaching the top: Strategies for closing academic achievement gaps. a report of the national study group for the affirmative development of academic ability. *North Central Regional Educational Laboratory*, 2004.
- [5] M. Bienkowski, M. Feng, and B. Means. Enhancing teaching and learning through educational data mining and learning analytics: An issue brief. *Office of Educational Technology, US Department of Education*, 2012.
- [6] E. L. Bjork, R. A. Bjork, et al. Making things hard on yourself, but in a good way: Creating desirable difficulties to enhance learning. *Psychology and the real world: Essays illustrating fundamental contributions to society*, 2(59-68), 2011.
- [7] J. S. Bruner and C. C. Goodman. Value and need as organizing factors in perception. *The journal of abnormal and social psychology*, 42(1):33, 1947.
- [8] P. C. Burnett and V. Mandel. Praise and feedback in the primary classroom: Teachers’ and students’ perspectives. *Australian Journal of Educational & Developmental Psychology*, 10:145–154, 2010.
- [9] R. Butler. Enhancing and undermining intrinsic motivation: The effects of task-involving and ego-involving evaluation on interest and performance. *British journal of educational psychology*, 58(1):1–14, 1988.
- [10] C. Candel, E. Vidal-Abarca, R. Cerdán, M. Lippmann, and S. Narciss. Effects of timing of formative feedback in computer-assisted learning environments. *Journal of Computer Assisted Learning*, 36(5):718–728, 2020.
- [11] A. K. Clark and P. Whetstone. The impact of an online tutoring program on mathematics achievement. *The Journal of Educational Research*, 107(6):462–466, 2014.
- [12] J. Cohen. *Statistical power analysis for the behavioral sciences*. Academic press, 2013.
- [13] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4(4):253–278, 1994.
- [14] K. P. Cross. Feedback in the classroom: Making assessment matter. ERIC, 1988.
- [15] S. Eastman and N. Kaeding. Opportunity zones: What we know and what we don’t. *Tax Foundation Fiscal Fact*, 630, 2019.
- [16] M. Feng, J. Roschelle, N. Heffernan, J. Fairman, and R. Murphy. Implementation of an intelligent tutoring system for online homework support in an efficacy trial. In *International Conference on Intelligent Tutoring Systems*, pages 561–566. Springer, 2014.
- [17] A. Flores. Examining disparities in mathematics education: Achievement gap or opportunity gap? *The High School Journal*, 91(1):29–42, 2007.
- [18] L. Fries, M. S. DeCaro, and G. Ramirez. The lure of seductive details during lecture learning. *Journal of Educational Psychology*, 111(4):736, 2019.
- [19] D. A. Gilman. Comparison of several feedback methods for correcting errors by computer-assisted instruction. *Journal of Educational Psychology*, 60(6p1):503, 1969.
- [20] N. T. Heffernan and C. L. Heffernan. The assistments ecosystem: Building a platform that brings scientists and teachers together for minimally invasive research on human learning and teaching. *International Journal of Artificial Intelligence in Education*, 24(4):470–497, 2014.
- [21] A. T. Jersild. Examination as an aid to learning. *Journal of Educational Psychology*, 20(8):602, 1929.
- [22] J. D. Karpicke. Retrieval-based learning: A decade of progress. *Grantee Submission*, 2017.
- [23] A. N. Kluger and A. DeNisi. The effects of feedback interventions on performance: A historical review, a meta-analysis, and a preliminary feedback intervention theory. *Psychological bulletin*, 119(2):254, 1996.
- [24] D. R. Krathwohl. A revision of bloom’s taxonomy: An overview. *Theory into practice*, 41(4):212–218, 2002.
- [25] N. J. Krichevsky and K. P. Spinelli. E-trials: Developing a web application for educational research. 2020.
- [26] C.-L. C. Kulik and J. A. Kulik. Effects of ability grouping on secondary school students: A meta-analysis of evaluation findings. *American educational research journal*, 19(3):415–428, 1982.
- [27] P. McGuire, S. Tu, M. E. Logue, C. A. Mason, and K. Ostrow. Counterintuitive effects of online feedback in middle school math: results from a randomized controlled trial in assistments. *Educational Media International*, 54(3):231–244, 2017.
- [28] K. V. Middleton. The longer-term impact of covid-19 on k–12 student learning and assessment. *Educational Measurement: Issues and Practice*, 39(3):41–44, 2020.
- [29] M. Montessori. *The advanced Montessori method*, volume 1. Frederick A. Stokes Company, 1917.
- [30] B. F. T. Moreira, T. S. S. Pinto, D. S. V. Starling, and A. Jaeger. Retrieval practice in classroom settings: a review of applied research. In *Frontiers in Education*, page 5. Frontiers, 2019.
- [31] R. Moreno. Decreasing cognitive load for novice students: Effects of explanatory versus corrective feedback in discovery-based multimedia. *Instructional science*, 32(1):99–113, 2004.
- [32] S. Narciss, S. Sosnovsky, L. Schnaubert, E. Andrès, A. Eichelmann, G. Gogvadze, and E. Melis. Exploring feedback and student characteristics relevant for personalizing feedback strategies. *Computers & Education*, 71:56–76, 2014.
- [33] T. Nguyen. The effectiveness of online learning: Beyond no significant difference and future horizons. *MERLOT Journal of Online Learning and Teaching*, 11(2):309–319, 2015.
- [34] K. Ostrow and N. Heffernan. Testing the multimedia principle in the real world: a comparison of video vs. text feedback in authentic middle school math assignments. In *Educational Data Mining 2014*, 2014.
- [35] J. M. Parr and L. Limbrick. Contextualising practice: Hallmarks of effective teachers of writing. *Teaching and Teacher Education*, 26(3):583–590, 2010.

- [36] P. I. Pavlik Jr, H. Cen, and K. R. Koedinger. Performance factors analysis—a new alternative to knowledge tracing. *Online Submission*, 2009.
- [37] A. G. Picciano, J. Seaman, P. Shea, and K. Swan. Examining the extent and nature of online learning in american k-12 education: The research initiatives of the alfred p. sloan foundation. *The internet and higher education*, 15(2):127–135, 2012.
- [38] E. Prihar, A. Botelho, M. Corace, A. Shanaj, Z. Dia, and N. T. Heffernan. Student engagement during remote learning. In *Companion Proceedings 11th International Conference on Learning Analytics Knowledge*, pages 49–51, 2021.
- [39] E. Prihar, T. Patikorn, A. Botelho, A. Sales, and N. Heffernan. Toward personalizing students’ education with crowdsourced tutoring. In *Proceedings of the Eighth ACM Conference on Learning@ Scale*, pages 37–45, 2021.
- [40] L. Razzaq and N. T. Heffernan. Scaffolding vs. hints in the assistment system. In *International Conference on Intelligent Tutoring Systems*, pages 635–644. Springer, 2006.
- [41] L. M. Razzaq, M. Feng, G. Nuzzo-Jones, N. T. Heffernan, K. R. Koedinger, B. Junker, S. Ritter, A. Knight, E. Mercado, T. E. Turner, et al. Blending assessment and instructional assisting. In *AIED*, pages 555–562, 2005.
- [42] L. M. Razzaq and N. T. Heffernan. To tutor or not to tutor: That is the question. In *AIED*, pages 457–464, 2009.
- [43] W. Roper. Feedback in computer assisted instruction. *Programmed learning and educational technology*, 14(1):43–49, 1977.
- [44] D. Selent, T. Patikorn, and N. Heffernan. Assistments dataset from multiple randomized controlled experiments. In *Proceedings of the Third (2016) ACM Conference on Learning@ Scale*, pages 181–184, 2016.
- [45] M. H. Smits, J. Boon, D. M. Sluijsmans, and T. Van Gog. Content and timing of feedback in a web-based learning environment: Effects on learning as a function of prior knowledge. *Interactive Learning Environments*, 16(2):183–193, 2008.
- [46] N. Sundararajan and O. Adesope. Keep it coherent: A meta-analysis of the seductive details effect. *Educational Psychology Review*, 32(3):707–734, 2020.
- [47] P. Suppes et al. Computer-assisted instruction: Stanford’s 1965-66 arithmetic program. 1968.
- [48] L. Voerman, P. C. Meijer, F. A. Korthagen, and R. J. Simons. Types and frequencies of feedback interventions in classroom interaction in secondary education. *Teaching and teacher education*, 28(8):1107–1115, 2012.
- [49] B. J. Zimmerman and D. H. Schunk. *Educational psychology: A century of contributions: A Project of Division 15 (educational Psychology) of the American Psychological Society*. Routledge, 2014.

Optimizing Representations and Policies for Question Sequencing using Reinforcement Learning

Aqil Zainal Azhar
University of Edinburgh
Z.A.B.Zainal-
Azhar@sms.ed.ac.uk

Avi Segal
Ben-Gurion University
avise@post.bgu.ac.il

Kobi Gal
Ben-Gurion University
University of Edinburgh
kobig@bgu.ac.il

ABSTRACT

This paper studies the use of Reinforcement Learning (RL) policies for optimizing the sequencing of online learning materials to students. Our approach provides an end to end pipeline for automatically deriving and evaluating robust representations of students' interactions and policies for content sequencing in online educational settings. We conduct the training and evaluation offline based on a publicly available dataset of diverse student online activities used by tens of thousands of students. We study the influence of the state representations on the performance of the obtained policy and its robustness towards perturbations on the environment dynamics induced by stronger and weaker learners. We show that 'bigger may not be better', in that increasing the complexity of the state space does not necessarily lead to better performance, as measured by expected future reward. We describe two methods for offline evaluation of the policy based on importance sampling and Monte Carlo policy evaluation. This work is a first step towards optimizing representations when designing policies for sequencing educational content that can be used in the real world.

1. INTRODUCTION

E-learning platforms have seen a surge in popularity over the last decade [10], spurred on by the increased Internet penetration into developing communities [4]. The target demographic has expanded beyond casual users/students as more organizations adopt e-learning to train their workforce and actively engage them in life-long learning [34].

As online educational settings become ubiquitous, there is a growing need for a personalized sequencing of content/support that can adapt to the individual differences of the student as well as their evolving pedagogical requirement throughout the course progression [28]. Research in cognitive science has long demonstrated the strong correlation between adapted material sequencing and learning outcomes [26]. Static e-learning platforms lack the capacity to respond to a student's 'cognitive state' and therefore perform poorly relative

to a human tutor [34].

Reinforcement learning (RL) offer a potential approach for adapting a learning sequence to students [9]. A learning sequence can be optimized based on a numerical reward (for example test marks) by a pedagogical agent that prescribes actions (adaptive feedback or sequencing of content) based on different states (approximated users' cognitive states).

There are two main challenges to using RL 'out of the box' in educational settings. First, how to choose the best representation to model student behavior? On the one hand, increasing the granularity dimension of the state-space allows to capture intricate dynamics in the model such as students' cognitive states and skills. On the other hand, models with complex state spaces are inherently more difficult to learn and the resulting policy may lack support in the data for parts of the state space. Second, how to evaluate the resulting sequencing policy? Ideally, sequencing policies will be deployed online and evaluated with real learners. This is costly or not technically feasible to carry out in many cases and an imperfect policy may adversely affect students' learning.

This paper addresses both of these challenges in the context of a new publicly available dataset containing the online interactions of thousands of students [7]. Our approach provides an end to end pipeline for automatically deriving and evaluating robust representations of students' interactions and policies for content sequencing in online educational settings.

To address the first challenge, we present a new greedy procedure to augment the representation space, by incrementally adding new features and choosing the best performing representations on held out data. Each policy is evaluated using expected cumulative reward. We provide several key insights about the use of RL in Educational contexts. First, that 'bigger is not always better', in that more complex state spaces may not always lead to better policy performance. Second, that including a 'forgetting' element in the state space, which is known to affect students' learning, significantly improved performance. Third, that strongly penalizing rewards from unseen state-action pairs in the data, can increase the support of the resulting policy without reducing performance.

To address the second challenge, we use two existing offline

A. Z. Azhar, A. Segal, and K. Gal. Optimizing representations and policies for question sequencing using reinforcement learning. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 39–49, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853123>

policy evaluation methods to reliably estimate the performance of the resulting policy using only the collected data. The first method uses importance sampling to correct for the difference in distributions between the learned policy and the policy that was used to gather the data. The second method simulates the learned policy using Monte Carlo methods. We also introduce a new approach that evaluates the policy with respect to perturbations induced by stronger and weaker learners. We find that perturbing the system dynamics have an adverse affect on performance of the learned policy, specifically in respect to weaker learners; and that increasing the granularity of the state-space makes the policy more robust to perturbations.

This work is a first step towards optimizing representations when designing policies for sequencing educational content that can be used in the real world. Our long term goal is to develop an adaptive RL based pedagogical agent that is able to optimize the sequence of learning materials (questions and lectures) to maximize student performance as measured by the expected ability to answer questions correctly at varying levels of difficulties. This agent will have the capacity to respond in real-time to a student’s current state as they progress through the learning material.

2. RELATED WORK

This paper relates to prior work in student modeling as well as automatically sequencing educational content to students [9]. A common approach in prior work is to integrate learning/cognitive theory into the construction of the student model. This imparts domain knowledge into the workflow and was shown to yield positive results by Doroudi et al. [9]. An example of such approach is the work by Bassen et al. [2]. Their objective was to optimize the sequencing of learning material from different knowledge components (KCs), to ‘maximize learning’. Their reward function is based on the difference between a post-test score (taken by users after completing the course) and a pre-test score (taken before the course). This metric is denoted as the Normalized Learning Gain (NLG). Training the agent with human participants is far too resource intensive. Instead their training is performed on a ‘simulated learner’ based on Bayesian Knowledge Tracing (BKT), a cognitive model that aims to estimate a learner’s mastery of different skills. The learner’s response to a particular question can be simulated based on the mastery of the related skill. The parameters of the BKT were set based on domain knowledge. Segal et al. [28] also utilised a similar cognitive based model, Item Response Theory (IRT) [14] to simulate student responses to questions of different level of difficulty. This is especially relevant since their objective was to sequentially deliver questions of differing levels of difficulty (rather than KCs) to maximize learning gains.

Other approaches forgo the framework of established learning models and instead manually design their simulators further integrating domain expertise. Dorcca et al. [8] employed a probabilistic model to simulate the learning process. Instead of sequencing activities by KCs or difficulties, they sequenced activities based on their associated learning style i.e. visual, verbal etc. Therefore, their simulator was manually designed based on research surrounding these principles. Iglesias et al. [15] utilised an expert derived artificial Markov Decision Process to act as the student model. This entails

manually describing the state space, transition probabilities and rewards. Similar to previous student models, this MDP can be used to simulate student responses to train the RL agent.

The works described so far do not utilise historical data (i.e., past interactions in the system and their results) to derive their student model. While integrating expert knowledge can be beneficial, a completely data-free proposition could impart strong biases. In our implementation, we take an alternative approach in using a purely data-driven model. There are existing literature which also do the same. For example, the authors of [30, 32, 5, 27] employed data-driven MDPs as their student model. Different than the hand-crafted MDP in Iglesias et al. [15], the transition probabilities and reward functions in these MDPs were obtained from the aggregated statistics observed in the dataset. Data-driven student models in literature were not only limited to data-driven MDPs. For instance, Beck et al. [3] utilised a linear regression model denoted as Population Student Model (PSM). PSM was trained on student trace data from a learning software and could simulate time taken and probability of a correct response.

Data-driven simulators require a quality training corpus that is sufficiently large and varied [30, 16]. In contrast to EdNet (a massive dataset collected over several years which we use in this work), the authors of previous papers were limited to much smaller scale datasets that were collected from a single cohort and could not evaluate their policy at scale. Our work is the first to provide an end to end pipeline from a large scale data source to a robust RL sequencing policy.

3. BACKGROUND

In this section we provide some necessary background in Reinforcement Learning & MDPs and briefly describe our dataset.

3.1 RL and MDPs

A Reinforcement Learning (RL) framework is governed by a Markov Decision Process [31, 17, 6] that is defined by the tuple of $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R})$, where $\mathcal{S} = s_1, \dots, s_n$ is the state space, $\mathcal{A} = a_1, \dots, a_m$ represents all available actions, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$ designates the transition probabilities between states conditioned on an action and $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$ denotes the reward function that is conditioned on the state, action and observed next state.

The goal of the agent is to maximize the **cumulative** reward it accumulates from each state. This cumulative reward is usually discounted by a factor γ raised to the power of t , to represent a lower perceived value for rewards received further in the future. The policy is a mapping of optimal actions to each state in \mathcal{S} . The discounted cumulative reward is denoted as the ‘return’, and the return from a particular state is associated with a policy π and the transition dynamics \mathcal{P} . The reward function ties the agent’s optimization goal with the modeller’s actual objective. Therefore its design must ensure those two criteria are aligned. The expected reward of a state (or the state value) under a deterministic policy

π can be defined as

$$V(s) = \mathbb{E}_{\pi, \mathcal{P}} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, \pi(t)) \mid s_0 = s \right] \quad (1)$$

where $\pi(t)$ is the action prescribed by the policy π at state t , $r(s_t, a_t)$ is the reward obtained in state s_t given action a , s_0 is the initial state and γ is a discount factor.

Model-based RL requires a model that holds information regarding the environment dynamics i.e. the transition probabilities \mathcal{P} and reward function \mathcal{R} . It is a common approach in domains where environment interactions are cost prohibitive [16][18]. The model mimics the behaviour of an environment and allows inferences about how the environment will respond to actions [31]. The real-world performance of the extracted policies are heavily influenced by the quality of the constructed model [16].

3.2 Data Description

EdNet [7] is a massive dataset of student logs from a MOOC learning platform in South Korea, *Santa*, collected by *Riiid! AI Research*¹. *Santa* covers a preparation course for the TOEIC (Test of English for International Communication) English proficiency exam. There are a total of 131,441,538 interactions collected from 784,309 students on the e-learning platform. These consist of user records of questions attempted, lectures watched and explanations reviewed, along with other meta information. EdNet logs are presented at 4 levels of hierarchy with higher levels providing higher fidelity logs, such as logs of lectures watched and the explanations reviewed. These are recorded in real time to provide an accurate chronological record of the students’ interaction with the platform. EdNet records detailed actions such as playing/pausing lectures and payment related information.

4. METHODOLOGY

Our approach, called GIFA (Greedy Iterative Feature Augmentation) maps a representation space of students’ activities to an optimal sequencing policy using RL. At each step, a representation-space for the education domain is defined using a set of features. An MDP model is defined over the representation space, and model-based RL is used to extract the optimal policy given the representation space. The policy is subsequently evaluated using the Expected Cumulative Reward metric. This process is iterated, greedily adding new features to the representation and computing the optimal policy given the representation (See top of Figure 1). The resulting representation and policy are verified using two offline policy evaluation processes (importance sampling and Monte Carlo) and the robustness of the policies is analyzed against perturbations corresponding to varying student types (See bottom of Figure 1). We proceed to describing our methodology in more detail.

4.1 State Space Representation

Several studies have shown the significant impact of the representation choice on RL performance, with some arguing it is just as influential as optimization algorithm itself [32, 29]. We design the representation candidates for our model to include features that are derived from the dataset. EdNet

¹<https://www.riid.co/>

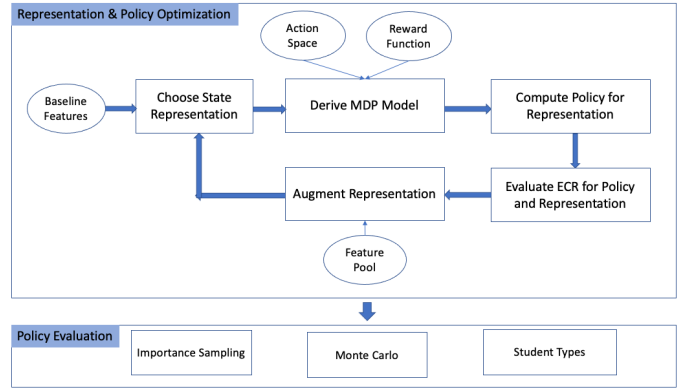


Figure 1: Pipeline of the GIFA approach

provides a total 13,169 questions and 1,021 lectures tagged with 293 types of skills [7]. Each question and lecture is segmented into one unique “part”, with 7 parts in total. Each part was grouped based on some meaningful domain criteria, such as a math topic [2]. EdNet offers a finer grouping of question/lectures according to the ‘skills’ (293 in total) they entail. Both ‘part’ and ‘skills’ grouping represent good candidates for the action space since they group the numerous questions/lectures in a domain meaningful manner. However the question bank is unevenly distributed with respect to the ‘skills’ grouping, which can lead to difficulties in getting equal support (or supporting observations) for each action type from the dataset. Therefore, ‘part’ was chosen on the basis that its 7 unique action space is both computationally feasible and better supported in the data.

A key aspect of modeling student learning is to represent the question difficulty in the state-space [28]. This enables to adapt the level of difficulty of a question to the student’s inferred skill level. Unfortunately EdNet does not directly classify questions/lectures into difficulty levels, meaning that they would need to be inferred from the student logs. A natural way to infer difficulty is to measure the percentage of correct answers submitted for a question and compare it with other questions in the question bank. A distribution over question difficulties can be created and quantiles can be derived to evenly split the questions into discrete levels of difficulty. Utilizing this process we created a difficulty level for each question, with the difficulty levels quantized into 4 levels ranging from 1 (easiest) to 4 (hardest).

To investigate the impact of the representation on performance, we create a feature pool, from which different representations are formed using the greedy iterative augmentation algorithm. The initial feature pool contains features that are widely seen in similar Reinforcement Learning driven Intelligent Tutoring implementations [2, 15, 32, 5]. The state features are longitudinal/temporal in nature so as to represent the users’ behaviour and performance over time. This sets a reasonable minimum requirement for the data gathering process, should this implementation be repeated with other datasets. The initial feature pool, their descriptions, and the associated granularity of their representations (bins), are shown in Table 1. Quantization of features to bins was performed so that a finite model can be formed.

Feature Pool	Bins	Description
"av time"	4	The cumulative average of the elapsed time measured at each activity.
"correct so far"	4	The ratio of correct responses to the number of activities attempted.
"prev correct"	3	A flag to indicate whether the user answered correctly in the previous question + fixed value for lecture.
"expl received"	4	Cumulative count of explanations reviewed by the user.
"steps-since-last"	8	A count of the number of steps since the current part was last encountered.
"lects consumed"	4	A cumulative count of the lectures consumed by a user.
"slow answer"	2	A flag to indicate whether the user's elapsed time for the preceding question was above the average elapsed time for that question.
"steps in part"	4	The cumulative count of how many steps a user has spent in the current part.
"avg fam"	4	The average part familiarity across all the 7 parts.
"topic fam"	4	Captures part familiarity of the previously chosen action (by amount of activities per topic).

Table 1: Initial Feature pool for state space

Relative to prior work that limited the feature size to be binary [2, 23, 32] our feature space is considerably larger. With each user covering on average 440 activities during their learning period within this dataset[7], a binary split would lose a lot of information on the evolution of a feature value throughout the course. Our features in contrast have up to 8 bins. This ultimately imposes a necessity for a quality training corpus that can provide sufficient support for each of the many unique combinations within the feature space. This is where the scale of the EdNet dataset provides a distinct advantage relative to previous implementations.

4.2 Deriving the student model

In this section we describe the derivation of an MDP-based student model using a selected set of features. We model the transition probabilities as multinomial distributions derived from state transition counts observed in the dataset as shown in Equation 2. This means that a particular outcome s_i , of enacting action a_k in state s_j has a probability given by the number of times that outcome was observed in the dataset, normalized by the sum of all possible outcomes observed under the same conditions.

$$\hat{p}(s_i | s_j, a_k) = \frac{c(s_i, s_j, a_k)}{\sum_{i=1}^n c(s_i, s_j, a_k)} \quad (2)$$

Where $c(s_i, s_j, a_k)$ is the count of observed transitions where enacting action a_k in state s_j leads to next state s_i . This provides the transition probabilities component of the MDP

student model for each (s, a, s') or the three argument dynamic [31].

In many standard MDP definitions [32, 30, 5], the reward function is also defined in terms of the three argument dynamic i.e. $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$. This assumes a deterministic environment reward with respect to a given (s, a, s') . The reward for a student's response depends on the question difficulty such that correct responses on harder levels indicate stronger performance and attain higher rewards. Inversely, incorrect responses on easier levels attain a larger punishment (negative reward). A symmetrical reward function was designed with rewards for questions ranging from 1 to 4 if answered correctly or -1 to -4 if answered incorrectly. We doubled the penalty of incorrect answers to achieve a normal distribution usually exhibited in student grades [13]. Thus rewards take values $r \in \{-8, -6, -4, -2, 0, 1, 2, 3, 4\}$. While the level of difficulty is captured by the action in the (s, a, s') tuple, the correctness is only captured in the states when 'prev_correct' (refer table 1) is included in the representation. Lectures do not have a correct/incorrect responses and so a default reward of '0' is assigned for lecture viewing actions.

The specific dynamic values are of course dependent on the actual state representations. As we continuously augment the representations (see next section), the support for each unique state will inevitably fall due to further division of the observations. Another factor in providing a balanced distribution of support within the transitions, is the variety of actions chosen in each state. This ultimately depends on the action space described earlier and the behaviour policy used to obtain the dataset. A higher fidelity action space will lead to an increase in the size transition space i.e. the unique combinations of (s, a, s') . Although unknown in most cases, it is important that the behaviour policy is sufficiently varied in terms of its action choices to ensure a balanced distribution of support. Because of this, a random behaviour policy fits the objective well [30]. Since users in EdNet are allowed to select the 'part' and the type of activity they work on [7], we make an assumption that this random criteria is partially fulfilled. The caveat here is that not all users have access to all parts i.e. free users are limited to parts 2 and 5 only.

4.3 Representation Selection

Key to designing a successful model of student behavior depends on deriving information on the student's cognitive state, which is a latent variable in the model. With more features in the representation, one should expect a better approximation of the students cognitive state and consequently a better equipped pedagogical agent to provide effective sequencing.

We utilise the GIFA approach in obtaining an optimal representation. This involves a search of the feature space and generation of several candidate feature subsets. Each of these subsets is evaluated based on its corresponding policy derived from a standard policy-iteration RL solution. The pseudocode for this process is shown in Algorithm 1. Note the limit on the number of features \mathcal{N} which can be based on a computational limit or a threshold of minimum support for every unique combination in the feature space.

Algorithm 1 Greedy Iterative Feature Augmentation algorithm

Input: Feature pool Ω , Dataset \mathcal{D} , Max. number of features \mathcal{N} (optional)
Set: Optimal feature representation \mathcal{S}^* .
while $size(\mathcal{S}^*) \leq \mathcal{N}$ **do**
 for $\omega_i \in \Omega$ **do**
 Set: $\mathcal{S}_i = \mathcal{S}^* + \omega_i$
 $MDP = Construct_MDP(\mathcal{S}_i, \mathcal{D})$
 $\pi^* = Policy_Iteration(MDP)$
 $ECR_i = Calculate_ECR(\pi^*)$
 end for
 Set $\mathcal{S}^* = \mathcal{S}_i$ with highest ECR_i .
 Remove feature from pool $\Omega = \Omega - \omega_i$
end while

Round	Rep.	ECR	ECR Diff. (%)	Rep. Size
Base	MDP_B	238.44	-	64
1	MDP_1	283.63	18.95	256
2	MDP_2	387.42	36.6	2048
3	MDP_3	392.05	1.19	4096
4	MDP_4	396.00	1.01	16384
5	MDP_5	396.00	0	65536

Table 2: ECR results showing best performing representation at each iteration

While our search procedure involves exhaustively looping through every remaining feature in the feature pool Ω to form the subsets, one can alternatively employ a different search algorithm such Monte Carlo Tree Search [12] or correlation based feature selection [29] to create more informed subsets that are likely to be better candidates. These techniques would be useful in limiting the number of iterations needed in larger feature pools and are left for future work.

As part of the greedy feature augmentation algorithm above, an evaluation metric for π^* , the computed policy, is required for each representation at every iteration. A common practice used to evaluate a given policy in RL is to use the Expected Cumulative Reward (ECR) metric which is the average of the expected cumulative reward under the policy π^* , across all initial states in the dataset ($V_{\pi^*}(s_0)$).

Note that in each state representation, the initial state of every user in the EdNet is the same. This is because we lack any prior information of the user before they begin to solve questions. When such information is available, the initial state can capture information from the students pre-test scores and so would vary across the students in the dataset.

5. RESULTS

A summary of the results from the greedy iterative augmentations is provided in Table 2. The feature description for the corresponding MDP representations are given in table 3. The base MDP is denoted as MDP_B. The ‘ECR Diff.’ column shows the percent improvement in ECR relative to the smaller representation preceding it. The ‘Rep. Size’ column illustrates the size of the state feature space. This is depen-

Representation	Features
MDP_B	topic_fam, correct_so_far, av_time
MDP_1	topic_fam, correct_so_far, av_time, expl_received
MDP_2	topic_fam, correct_so_far, av_time, expl_received, ssl
MDP_3	topic_fam, correct_so_far, av_time, expl_received, ssl, prev_correct
MDP_4	topic_fam, correct_so_far, av_time, expl_received, ssl, prev_correct, av_fam
MDP_5	topic_fam, correct_so_far, av_time, expl_received, ssl, prev_correct, av_fam, time_in_part

Table 3: Representation outputted by algorithm for each round (added features highlighted)

dent on each constituent feature’s bin size, i.e. the number of discrete bins allocated. Note that results presented here are only showing the best performing representation at each round of the feature augmentation.

5.1 ECR Analysis

The largest spike in ECR followed at the second round of augmentations with the addition of the ‘steps-since-last’ (ssl) feature with an increase of 36.6% over the preceding representation. This feature measures the number of steps or activities (questions/lectures) consumed since the current part was last encountered. This feature is inferring the ‘forgetting’ element during the learning process and was inspired by the ‘spacing effect’ described in [11].

Early research in instructional sequencing in language learning used models of forgetting to great success [1]. Our findings concur with this, in that by including ‘ssl’ into the feature space, we dramatically increased the policy performance. One could argue that this ECR increase was more influenced by the larger bin allocation to ‘ssl’ (8 relative to 4 for most other features) rather than the actual utility of the domain information it is measuring. However, if that were the case, then we would expect ‘ssl’ to be the first feature added to the base representation. This was not the case since the best performing feature in the first round of augmentations was ‘expl_received’, a 4-size bin feature. Nonetheless, further exploration is needed to further learn the influence of bin sizes on the results.

At the final round of iteration, the performance of representation MDP_5 only equals the performance of preceding representation MDP_4. Though we did not have a specified limit imposed on the number of features, \mathcal{N} , the performance plateau exhibited at this final round indicated a suitable termination point for the augmentation algorithm. And since MDP_4 produced equal performance to MDP_5 with a smaller representation size, it was chosen to be the optimum representation within this feature pool.

5.2 Correcting for OOD actions

This section describes our technique for handling the uncertainty induced by unseen or out of distribution (OOD) actions in the RL policy. Some of the extracted policies in the GIFA algorithm included state-actions pairs with 0 support from the training data. Such behaviour is deemed

unsafe for a computed policy [18, 20] as the resulting reward from such combinations of unsupported state-action pairs is unexpected. Specifically, this situation should not occur under policies derived from tabular methods as explained in section 3. Nonetheless we discovered that some of the larger representations yielded policies with unseen actions. The policy derived from MDP_4 prescribed unseen actions in 10 states. While this was a small fraction of the total state space (around 65,000), unseen actions are an important issue to address because, in the tabular case, any state-action value estimates must be derived only from related experiences [31].

By default, our algorithm prescribes state-action pairs a value of zero if it was never observed. We discovered that the problematic states themselves had very little support in the dataset and were only observed transitioning to themselves, before the episode ends. In the few times the state was visited, a negative or zero reward was produced. Since these states would only transition to themselves, the values of these valid actions were either negative or zero. Hence, from the algorithms perspective, an invalid (unseen) action with a default value of zero, was preferable (or equal) to the observed actions.

To combat this issue, we modified the MDP representations to strongly penalise the rewards from unseen state-action pairs, in the form of a -9999 reward. This discouraged the policy from choosing such actions even if the only valid actions yielded zero or negative returns (The worse case is bounded from below at $\sum_{k \rightarrow \infty} \gamma^k \times (-8)$ and is clearly higher than $\sum_{k \rightarrow \infty} \gamma^k \times (-9999)$). With these changes in place, we observe no unseen actions in any of the policies. The performance rank of representations remained constant with the ECR changes almost negligible. This is because the states involved were observed very infrequently and occupied a small probability mass in the transition probabilities. We note that the fix did not have a statistically significant effect on our results. In the analysis that follows we will utilise the penalised representations. We also note that research by Liu et al. [20] implemented a ‘pessimistic policy iteration’ approach that similarly penalises insufficiently supported state-action tuples (filtered by a threshold).

6. OFFLINE POLICY EVALUATION

Ideally, the outputted policy from the GIFA approach would be evaluated in real time with students who directly interact in the EdNet environment. However, mistakes in the policy can have adverse affects on students’ engagements and learning [21]. The offline policy evaluation (OPE) field has been developed specifically to address this issue, by providing reliable estimates of policy performance using only past collected data [21]. We undertake two different OPE approaches to evaluate the computed optimal policies from the previous sections. Furthermore, we add a 3rd offline evaluation approach which evaluates the robustness of the computed policies to different model perturbations representing several student types.

6.1 Rollouts: Monte Carlo Policy Evaluation

The first OPE approach relies on the family of ‘Direct Methods’ for policy evaluation. These methods focus on regression based techniques to directly estimate the value function

of a policy under a given target policy [33]. Most of these methods do not need an estimation of the behaviour policy which was used to collect the dataset. In this method we implement a model-based direct method, Monte Carlo (MC) Policy Evaluation. This involves performing ‘rollouts’ from the initial states using the target policy until episode termination. The observed returns from each state in the rollout are averaged across many rollouts to yield the value function of the state. To ‘rollout’ our policy we would need to interact with the environment. However, as the name ‘model-based’ suggests, a model (in our case, the data-derived MDPs) acting as simulator allows us to perform these rollouts offline.

A key requirement for MC policy evaluation is an **episodic environment**, one where the episode terminates at a finite step at a ‘terminating state’ [31]. Although the user episodes in EdNet are finite, our data-derived MDPs are continuing i.e. without a terminating state. While a default terminating state could have easily been created for this purpose, it is unclear as to which ‘action’ would transition the final observed state to this terminating state and what ‘reward’ it would receive in the process. The choice of reward, could inadvertently impact the decisions made by the policy at the earlier states. Hence our MDPs were designed to be continuing to avoid this ambiguity.

This poses a problem with MC policy evaluation since the returns are only calculated when the episode ends. A potential workaround was to manually terminate the episode at a fixed length of rollout and calculate the returns from there. We chose a rollout length of 1000 steps and show that because of the discounting, any reward, $r \in \mathcal{R}$, received past this step, will have a negligible influence on the return of the initial state i.e. $\max_r (\gamma^{1000} ||r \in \mathcal{R}||) \approx 3.5 \times 10^{-4}$. Hence this rollout length provides a good approximation of the long term return, since any future actions will have minimal influence on the value of the initial state, i.e. the only state value of concern in our analysis. However, this assumption will only work with the ‘first-visit’ variant of MC policy evaluation (equation 3), where only the returns of a state when it was first encountered in the episode are considered and averaged across the rollouts [31]. This is opposed to the ‘every-visit’ variant which considers all the returns from a state every time it is visited in the episode. A fixed rollout length will not be suitable in the latter variant, since the initial state could be encountered more than once during the rollout. For example, if s_0 was encountered again at the 500th time step, then its return estimate for the second visit is based only on the remaining 500 future steps. Implementing the first-visit variant ensures that all $V(s_0)$ estimates are derived from observations spanning 1000 time steps ahead.

$$V_{\pi_e}^{MC}(s_0) = \frac{1}{N_{rollouts}} \sum_1^{N_{rollouts}} \sum_{t=1}^{1000} \gamma^t r_t^t \quad (3)$$

We now evaluate the policies under the MC Policy Evaluation method. A curve is plotted for the returns (cumulative rewards) from the initial state as the rollout progresses until the 1000th step for a total of 100 rollouts. The 95% confidence intervals are plotted around the mean value of the

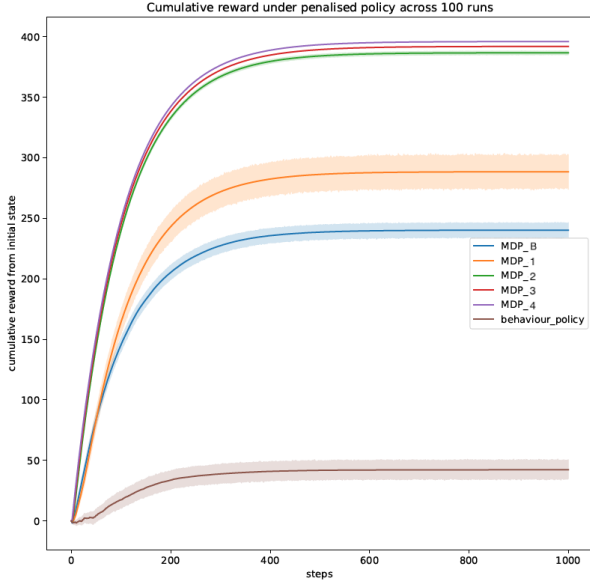


Figure 2: Monte Carlo Evaluation: Returns from s_0 across steps under the policies from the associated representations

rollouts. This analysis is shown in figure 2. As with the ECR, we can see that the improvements start to diminish significantly after the second round of augmentations. The performance of the estimated stochastic *behaviour policy* under this simulation (the policy that is directly induced from the data) is illustrated as a baseline. From this comparison, we see much better student performance under the RL policies than the baseline, signalling that the adaptive behaviour of the policy under RL framework is superior than the strategies used in the behaviour policy. We can also conclude that the larger representations exhibit better performance, potentially owing to a better approximation of the cognitive state as hypothesized.

6.2 Importance Sampling Policy Evaluation

The second OPE approach used relies on Importance Sampling. Importance sampling (IS) is a range of methods that in general estimate the expected values under one distributions given samples from another [31]. A wide range of RL literature have adopted this method as a way of evaluating a *target policy* (the policies derived from the RL algorithms) given samples derived from the *behaviour policy* (the policy used to gather the data) [16]. In this work we use the Weighted IS (WIS) metric presented in equation 4.

$$WIS = \frac{\sum_{i=1}^N [(\prod_{t=0}^{T_i} \frac{\pi_e(a_t^i | s_t^i)}{\pi_b(a_t^i | s_t^i)}) (\sum_{t=1}^{T_i} \gamma^t r_t^i)]}{\sum_{i=1}^N \prod_{t=0}^{T_i} \frac{\pi_e(a_t^i | s_t^i)}{\pi_b(a_t^i | s_t^i)}} \quad (4)$$

In this equation, N represents the number of users in the dataset and T_i is the trajectory length observed for user i .

A key feature in the formula is the **importance sampling ratio** $\prod_{t=0}^{T_i} \frac{\pi_e(a_t^i | s_t^i)}{\pi_b(a_t^i | s_t^i)}$, which considers the differences in ac-

Rep.	WIS
MDP_B	-4.146
MDP_1	-0.940
MDP_2	-4.382
MDP_3	4.319
MDP_4	4.910

Table 4: Importance sampling results - different representations

tion probabilities between the target policy π_e and the behaviour policy π_b . The product of the individual ratios across $t_i = 0 \rightarrow T$ quantifies whether a given sequence is more (or less) likely under π_e than π_b and therefore weights the returns accordingly. Averaging this across the entire dataset has the effect of adjusting the **expected return** sampled from the distribution generated by π_b to estimate the **expected return** sampled from π_e .

A well documented problem with IS estimators is the high variance induced by the importance sampling ratio due to: 1) a large difference between the two policies or 2) a long horizon length for the trajectories [33]. WIS reduces this variance by utilizing a weighted average instead of the simple average in standard Importance Sampling metric [31]. This method relies on the available knowledge of the behaviour policy. Since we do not have explicit information on this, π_b must be estimated from the dataset, \mathcal{D} as shown in equation 5.

$$\hat{\pi}_b(a|s) = \frac{\sum_{s,a \in \mathcal{D}} \mathbf{1}[s = s, a = a]}{\sum_{s \in \mathcal{D}} \mathbf{1}[s = s]} \quad (5)$$

The results of the WIS metric for the different representations are shown in table 4. We see that the 3 smaller representations yield negative values, indicating expected average failure in solving questions when utilizing these representations. For larger MDP representations, we can observe better performance, with MDP_4 demonstrating the best estimated performance.

6.3 Evaluating Model Robustness for Different Student Types

The optimal policy computed in the previous section relies on the average rewards and transition probabilities estimated from available data. In practice, these parameters are noisy and may change in different situations and during the execution of a policy [19]. As such, the performance of the computed policy may deteriorate significantly with changes in the environment dynamics [22]. In our case, with the MDP representing students acting in an educational system, this uncertainty represents the challenge of how to model parameters change for different student types and how do these changes influence the outcome of the computed policies. To model this uncertainty, we use a simplified robust MDP framework [25] where the uncertainty in model parameters is tied to specific student types. Specifically, we test the robustness of the computed policies under perturbations of the environment dynamics which are tied to two different student types. These perturbations are domain informed and are designed to correspond to ‘stronger’ and ‘weaker’ students types.

Algorithm 2 Domain informed perturbations

1: **Input:** Set of features to perturb $\bar{\Omega}$, MDP transition probabilities \mathcal{P}_{MDP} , set of domain filters for each feature ψ , positive perturbation constant $c = 0.05$
2: **for** $p_{s,a,s'} \in \mathcal{P}_{MDP}$ **do**
3: $\Delta_{s,a,s'} = \sum_{\omega \in \bar{\Omega}} \Delta_{\omega}$
 Where $\Delta_{\omega} = \begin{cases} c, & \text{if } \omega_s, \omega_{s'} \text{ satisfies } \psi_{\omega} \\ 0, & \text{else} \end{cases}$
4: **end for**
5: Adjust $\Delta_{s,a,s'}$ relative to others within the s, a pair:
 $\Delta_{s,a,s'} = \Delta_{s,a,s'} - \frac{1}{|\bar{\psi}|} \sum_{s'} \Delta_{s,a,s} \forall \Delta_{s,a,s'}$
6: Set perturbed transition probabilities $\bar{\mathcal{P}}_{MDP} = \mathcal{P}_{MDP}$
7: **for** $p_{s,a,s'} \in \bar{\mathcal{P}}_{MDP}$ **do**
8: $p_{s,a,s'} = \max(p_{s,a,s'} + \Delta_{s,a,s}, 0)$
9: **end for**
10: $p_{s,a,s'} = \frac{p_{s,a,s'}}{\sum_{s'} p_{s,a,s'}} \forall p_{s,a,s'}$
11: **Return:** $\bar{\mathcal{P}}_{MDP}$

Feature to Perturb	Strong	Weak
Topic_fam	$\omega_{s'} > \omega_s$	$\omega_{s'} = \omega_s$
Correct_so_far	$\omega_{s'} \geq \omega_s$	$\omega_{s'} < \omega_s$
Avg_time	$\omega_{s'} \leq \omega_s$	$\omega_{s'} > \omega_s$

Table 5: Domain perturbation filters, ψ for each feature in $\bar{\Omega}$ for the ‘Strong’ and ‘Weak’ perturbed MDPs, $\bar{\mathcal{P}}$ respectively

We now describe the perturbation process and analysis. We define a set of domain informed filters ψ for each perturbed feature. In our implementation we perturb three base features that were common in all representations i.e. ‘Topic_fam’, ‘Correct_so_far’ and ‘Avg_time’. The domain rules for the two separate perturbations ‘Strong’ and ‘Weak’ are defined in table 5. Specifically, in these perturbations we boost the topic familiarity and correctness and reduce the cumulative elapsed time for the ‘Strong’ student type and we reduce correctness and increase the cumulative elapsed time for the ‘Weak’ student type. For example, for the feature ‘Correct_so_far’ and the ‘Strong’ user case, we set the filter to capture transitions where the next state s' registers a greater or equal value relative to the current state s . When this filter is inputted in algorithm 2, the transitions that satisfy this filter will be boosted by the constant c . This ultimately has the effect of increasing the probability mass of this transition, perturbing the original MDP to make such transitions more likely.

Algorithm 2 introduces the perturbation process. In lines 2 to 4 we compute the transition probability deltas that are required for every transition which satisfies one or more perturbation filter. This is done for all state transitions in the MDP. In line 5 we create transition deltas for all transitions, accounting for the deltas introduced by the perturbed transitions. In lines 6 to 9 we apply the transition deltas to the transition probabilities, and finally in line 10 we normalize the transition probabilities following the changes made.

The results of two separate perturbations are measured by performing a policy evaluation algorithm with the original policy but under the perturbed MDPs (Strong & Weak) as the simulators.

Figure 3 shows the results of this analysis for the different representations. Notice that in all the representations, the original non-perturbed MDP always yielded the best performance (non-visible for MDP_4 as the student type lines overlap). This is expected, since the original policy was derived to perform optimally on the original MDP. However, as the representation size increases, the effects of the perturbations becomes less pronounced, almost becoming negligible past MDP_1. To determine if the larger representation would be affected with more features perturbed, we conducted another round of perturbations, this time only on MDP_4 and with **all** of its features (barring ‘ssl’) perturbed. The results show that the performance of the policy was not affected by the extended perturbations. This means that the MDP_4 is more robust towards deviations from the expected dynamics derived from the data. Hence, we have increased confidence that such policies would be robust in the real-world setting, maintaining their performance for students that exhibit different learning characteristics than those averaged over the observations in the EdNet dataset.

Taking the three offline evaluation results in combination we conclude that MDP_4 demonstrated the best performance across representations and perturbations.

7. DOMAIN RELATED INSIGHTS

By analyzing the state values and the policies derived from the RL algorithms, we can discover interesting insights in the way the policy behaves with respect to different learners. We demonstrate this approach on the simpler MDP_B which is based on the 3 features: topic familiarity, correct so far and average time. In figure 4, we plot the derived state values against the av_time and correct_so_far features in MDP_B. Based on our reward design, the state values indicates the future user performance. The expected future performance of the policy is much higher when the student has a high correct_so_far answer ratio. However, the relationship between the average time and state values is more complex. At higher values of ‘correct_so_far’, a higher ‘av_time’ entails a larger state value, but when ‘correct_so_far’ is low, the opposite is true and in such a case lower values of ‘av_time’ entails a larger state value. This means that for students with lower success so far, faster average time is indicative of higher future success. We hypothesize that this is due to the policy’s inability to significantly assist students which are consistently unsuccessful in solving questions and which are also taking relatively long time to dwell on each and every question. We note that even if the policies themselves are not used, findings like this can inform us of useful features and their relationships in predicting future user performance following informed interventions. Such findings can also inform us on the limits of automated approaches, and on the need for additional tailored support for struggling students, e.g. by supplying personalized human assistance where automated approaches are expected to demonstrate low effectiveness.

Analyzing the action choices in the policies, we discover that the RL algorithms tend to put preference on level 4 actions (harder questions). Indeed these do yield the highest reward and the lowest punishment in our reward design. One possible extension is to investigate how a change in the reward function design would impact the policy preferences.

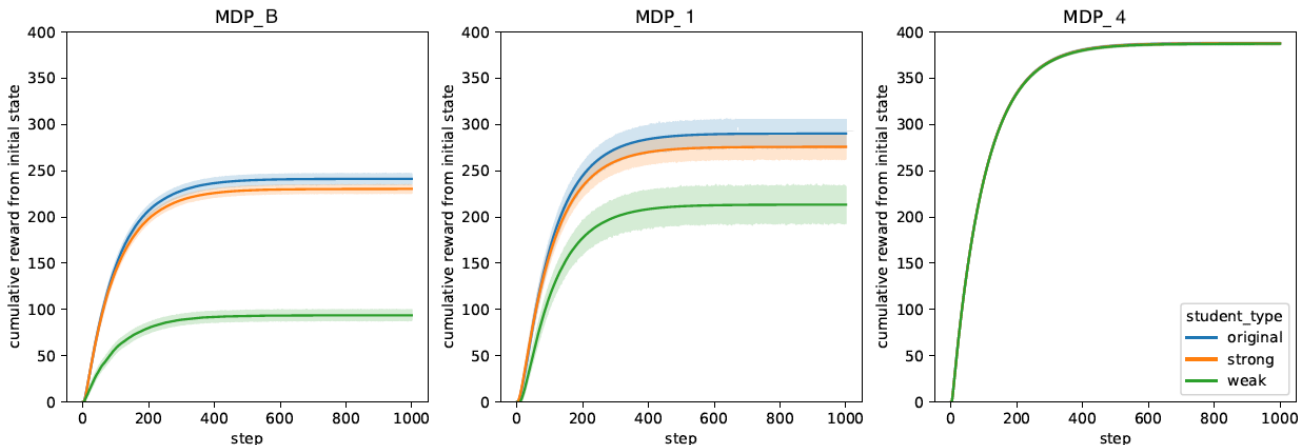


Figure 3: MC Policy Evaluation of the original policy under the perturbed MDPs

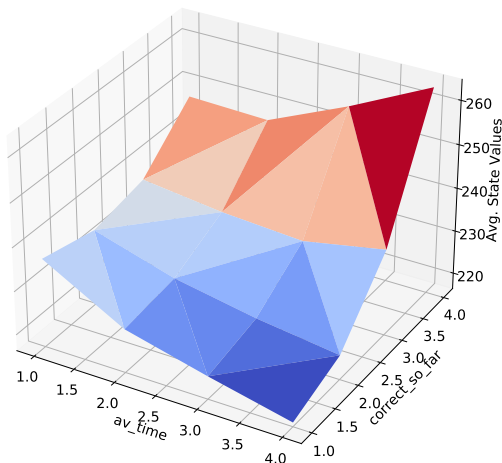


Figure 4: State values vs features

Again, even if the computed policies are not deployed in the field, such analysis can be useful as a technique in letting the data guide pedagogical strategies, for example by connecting pedagogically justified rewards to sequencing policies that maximize such a reward given the available data.

8. CONCLUSIONS AND FUTURE WORK

In this paper we approached the challenge of designing an adaptive RL based policy for optimizing the sequencing of learning materials to maximize learning. Human tutors usually outperform their computer counterparts, in that they are able to adapt to certain cues exhibited by the student during learning [34]. Training an RL policy with actual users is far too resource intensive. Therefore, we simultaneously tackle the problem of training and evaluating an RL algorithm offline based only on pre-collected data. A purely data-driven student model was created for this purpose. We hypothesized that a complex model is required to capture the intricacies of human learning. To investigate this theory, a large dataset, EdNet, was necessary to provide sufficient

support for the models.

Our student model was constructed in the form of a data-derived MDP, with the transition and reward dynamics estimated from the observations in the data. The raw logs were transformed into domain inspired features. By using the MDPs we then trained our agents with the model-based Policy Iteration algorithm. To determine whether a more complex model yields better tutoring, we employed a greedy iterative augmentation procedure. The ECR metric guided how we chose our features and demonstrated the positive relationship between representation complexity and policy performance. In our analyses we discovered issues with Out of Distribution actions in the policies and presented a solution in the form of penalising rewards. We further evaluated our policies using the Monte Carlo and Importance Sampling Policy Evaluation algorithms and tested the policies robustness against domain informed perturbations of the dynamics. We show that the larger representation are less impacted by the perturbations and therefore can provide a more equal learning experience for stronger or weaker students.

Several limitations are acknowledged which consequently open up further investigations. The influence of the bin-size on feature preference in the representations was discussed briefly but lacked conclusive evidence to rule out entirely. This work is necessary to ensure that the features are selected based only on the utility of the domain information they capture. From our model-based policy analyses we also discovered out-of-distribution actions in the policy space. Though we managed to remedy the problems for completely unseen actions through strong penalisation, the next course of action is to also penalise low supported actions/states **variably** according to their uncertainty as was explored by [20, 35]. We would also like to compare our approach to other feature selection and augmentation algorithms, such as genetic based metaheuristics [24]. Finally, the inferred policies should be evaluated in the real world in a controlled study.

9. ACKNOWLEDGEMENTS

This work was supported in part by the European Union Horizon 2020 WeNet research and innovation program under grant agreement No 823783

10. REFERENCES

- [1] R. C. Atkinson. Optimizing the learning of a second-language vocabulary. *Journal of experimental psychology*, 96(1):124, 1972.
- [2] J. Bassen, B. Balaji, M. Schaarschmidt, C. Thille, J. Painter, D. Zimmaro, A. Games, E. Fast, and J. C. Mitchell. Reinforcement learning for the adaptive scheduling of educational activities. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–12, 2020.
- [3] J. Beck, B. P. Woolf, and C. R. Beal. Advisor: A machine learning architecture for intelligent tutor construction. *AAAI/IAAI*, 2000(552-557):1–2, 2000.
- [4] J. Capper. E-learning growth and promise for the developing world. *TechKnowLogia*, 2(2):7–10, 2001.
- [5] M. Chi, K. VanLehn, and D. Litman. Do micro-level tutorial decisions matter: Applying reinforcement learning to induce pedagogical tutorial tactics. In *International conference on intelligent tutoring systems*, pages 224–234. Springer, 2010.
- [6] M. Chi, K. VanLehn, D. Litman, and P. Jordan. Inducing effective pedagogical strategies using learning context features. In *International conference on user modeling, adaptation, and personalization*, pages 147–158. Springer, 2010.
- [7] Y. Choi, Y. Lee, D. Shin, J. Cho, S. Park, S. Lee, J. Baek, C. Bae, B. Kim, and J. Heo. Ednet: A large-scale hierarchical dataset in education. In *International Conference on Artificial Intelligence in Education*, pages 69–73. Springer, 2020.
- [8] F. A. Dorça, L. V. Lima, M. A. Fernandes, and C. R. Lopes. Comparing strategies for modeling students learning styles through reinforcement learning in adaptive and intelligent educational systems: An experimental analysis. *Expert Systems with Applications*, 40(6):2092–2101, 2013.
- [9] S. Doroudi, V. Alevan, and E. Brunskill. Where’s the reward? *International Journal of Artificial Intelligence in Education*, 29(4):568–620, 2019.
- [10] E. Duffin. E-learning and digital education-statistics & facts. Retrieved December, 22:2019, 2019.
- [11] H. Ebbinghaus. *Über das gedächtnis: untersuchungen zur experimentellen psychologie*. Duncker & Humblot, 1885.
- [12] R. Gaudel and M. Sebag. Feature selection as a one-player game. In *International Conference on Machine Learning*, pages 359–366, 2010.
- [13] T. Grosgees and D. Barchiesi. European credit transfer and accumulation system: An alternative way to calculate the ects grades. *Higher Education in Europe*, 32(2-3):213–227, 2007.
- [14] R. K. Hambleton, R. J. Shavelson, N. M. Webb, H. Swaminathan, and H. J. Rogers. *Fundamentals of item response theory*, volume 2. Sage, 1991.
- [15] A. Iglesias, P. Martínez, R. Aler, and F. Fernández. Reinforcement learning of pedagogical policies in adaptive and intelligent educational systems. *Knowledge-Based Systems*, 22(4):266–270, 2009.
- [16] S. Ju, S. Shen, H. Azizzoltani, T. Barnes, and M. Chi. Importance sampling to identify empirically valid policies and their critical decisions. In *EDM (Workshops)*, pages 69–78, 2019.
- [17] H. Le, C. Voloshin, and Y. Yue. Batch policy learning under constraints. In *International Conference on Machine Learning*, pages 3703–3712. PMLR, 2019.
- [18] S. Levine, A. Kumar, G. Tucker, and J. Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.
- [19] S. H. Lim, H. Xu, and S. Mannor. Reinforcement learning in robust markov decision processes. *Advances in Neural Information Processing Systems*, 26, 2013.
- [20] Y. Liu, A. Swaminathan, A. Agarwal, and E. Brunskill. Provably good batch off-policy reinforcement learning without great exploration. *Advances in Neural Information Processing Systems*, 33:1264–1274, 2020.
- [21] T. Mandel, Y.-E. Liu, S. Levine, E. Brunskill, and Z. Popovic. Offline policy evaluation across representations with applications to educational games. In *AAMAS*, volume 1077, 2014.
- [22] S. Mannor, D. Simester, P. Sun, and J. N. Tsitsiklis. Bias and variance approximation in value function estimates. *Management Science*, 53(2):308–322, 2007.
- [23] Y. Mao. One minute is enough: Early prediction of student success and event-level difficulty during novice programming tasks. In *In: Proceedings of the 12th International Conference on Educational Data Mining (EDM 2019)*, 2019.
- [24] S. Mirjalili. Genetic algorithm. In *Evolutionary algorithms and neural networks*, pages 43–55. Springer, 2019.
- [25] A. Nilim and L. El Ghaoui. Robust control of markov decision processes with uncertain transition matrices. *Operations Research*, 53(5):780–798, 2005.
- [26] F. E. Ritter, J. Nerb, E. Lehtinen, and T. M. O’Shea. *In order to learn: How the sequence of topics influences learning*. Oxford University Press, 2007.
- [27] J. Rowe, B. Pokorny, B. Goldberg, B. Mott, and J. Lester. Toward simulated students for reinforcement learning-driven tutorial planning in gift. In *Proceedings of R. Sottolare (Ed.) 5th Annual GIFT Users Symposium. Orlando, FL*, 2017.
- [28] A. Segal, Y. B. David, J. J. Williams, K. Gal, and Y. Shalom. Combining difficulty ranking with multi-armed bandits to sequence educational content. In *International conference on artificial intelligence in education*, pages 317–321. Springer, 2018.
- [29] S. Shen and M. Chi. Aim low: Correlation-based feature selection for model-based reinforcement learning. *International Educational Data Mining Society*, 2016.
- [30] S. Shen and M. Chi. Reinforcement learning: the sooner the better, or the later the better? In *Proceedings of the 2016 conference on user modeling adaptation and personalization*, pages 37–44, 2016.
- [31] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.
- [32] J. R. Tetreault and D. J. Litman. A reinforcement

- learning approach to evaluating state representations in spoken dialogue systems. *Speech Communication*, 50(8-9):683–696, 2008.
- [33] C. Voloshin, H. M. Le, N. Jiang, and Y. Yue. Empirical study of off-policy policy evaluation for reinforcement learning. *arXiv preprint arXiv:1911.06854*, 2019.
- [34] B. P. Woolf. *Building intelligent interactive tutors: Student-centered strategies for revolutionizing e-learning*. Morgan Kaufmann, 2010.
- [35] T. Yu, G. Thomas, L. Yu, S. Ermon, J. Zou, S. Levine, C. Finn, and T. Ma. Mopo: Model-based offline policy optimization. *arXiv preprint arXiv:2005.13239*, 2020.

Code-DKT: A Code-based Knowledge Tracing Model for Programming Tasks

Yang Shi, Min Chi, Tiffany Barnes, Thomas W. Price
North Carolina State University
Raleigh, NC, USA
{yshi26, mchi, tmbarnes, twprice}@ncsu.edu

ABSTRACT

Knowledge tracing (KT) models are a popular approach for predicting students' future performance at practice problems using their prior attempts. Though many innovations have been made in KT, most models including the state-of-the-art Deep KT (DKT) mainly leverage each student's response either as correct or incorrect, ignoring *its content*. In this work, we propose **Code-based Deep Knowledge Tracing (Code-DKT)**, a model that uses an attention mechanism to automatically extract and select domain-specific code features to extend DKT. We compared the effectiveness of Code-DKT against Bayesian and Deep Knowledge Tracing (BKT and DKT) on a dataset from a class of 50 students attempting to solve 5 introductory programming assignments. Our results show that Code-DKT consistently outperforms DKT by 3.07 – 4.00% AUC across the 5 assignments, a comparable improvement to other state-of-the-art domain-general KT models over DKT. Finally, we analyze problem-specific performance through a set of case studies for one assignment to demonstrate when and how code features improve Code-DKT's predictions.

Keywords

Knowledge Tracing, Deep Knowledge Tracing, CS Education, Code Analysis, Deep Learning

1. INTRODUCTION

Modeling student knowledge to predict performance on future problems, called Knowledge Tracing (KT), is a fundamental feature of intelligent tutoring systems [50]. KT models enable tutoring systems to support mastery learning [14], select appropriate next problems [1], provide help [46], and provide analytics to instructors [34], all of which can improve learning. KT models have increased in complexity from the early 4-parameter Bayesian Knowledge Tracing (BKT) to modern models that train deep neural networks with tens of thousands of parameters using the latest deep learning innovations (e.g. attention [54] and transformers [51]). This has

led to improvement in KT model performance, especially for larger datasets, e.g. from ASSISTments [41, 17].

The *simplest version* of the KT problem uses only the sequence of: 1) which problems the student has attempted, and 2) whether or not each attempt was correct. While this makes KT models widely applicable across domains, this also omits a potential wealth of information about *how* the student attempted each problem. Increasingly, ITS being built to support complex problem solving tasks, like programming in Snap [35] and in games [22], logic proofs [30], science inquiry [25] and language learning [47]. In these domains, correctness may not provide enough information about student knowledge, varying significantly in the reasons both for incorrectness and correctness. In programming, for example, one incorrect attempt may have a minor syntax error while another includes a clear misconception. Similarly, two different *correct* answers could reveal dramatically different levels of concept mastery depending on their conciseness and the concepts used. Most KT models would treat all correct and all incorrect attempts identically. A domain-specific KT model, e.g. those for science by Rowe et al. [40], might greatly improve KT performance. Little work has investigated whether domain-general KT models can predict student success in programming, or how domain-specific features might improve performance.

In this paper, we explore when and how features extracted from students' submitted code can improve a KT model for programming. To do so, we introduce a novel code-based deep knowledge tracing (Code-DKT) model, which uses the code2vec model [4] to learn a meaningful representation of student code, and combines this with Deep Knowledge Tracing (DKT) [34] to track student progress. Specifically, student code submissions are represented with abstract syntax trees, and split into multiple code paths [4] (explained in Section 3). We assign the importance of different code paths by learning weights guided by the scores students received for the current and past submissions. We compared the performance of Code-DKT with baseline BKT and DKT models on a dataset of 50 introductory programming problems from 410 students, across 5 assignments. Our experiments show that the Code-DKT model is able to consistently improve DKT's performance by 3.08-4.00 percentage points in AUC. This improvement is comparable to that of other modern KT models over DKT (2-4%) [31, 44], suggesting that domain-specific features may be just as important as model structure. Finally, we investigate one assignment through 3

Y. Shi, M. Chi, T. Barnes, and T. Price. Code-DKT: A code-based knowledge tracing model for programming tasks. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 50–61, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853105>

case studies to explore the mechanisms by which code features may improve the model, and when they are most useful. We also show that Code-DKT outperforms more naive code-feature models. Overall, this paper makes three contributions: 1) the Code-DKT model, which extends DKT for programming tasks; 2) evidence that Code-DKT outperforms both domain-general models and naive code-feature models; and 3) evidence of when and how Code-DKT’s code features improve model performance.

2. RELATED WORK

In this section, we present related work on knowledge tracing, student modeling in computer science education, and deep learning models for code/programs.

2.1 Knowledge Tracing

Knowledge tracing (KT) models student knowledge as they solve problems to predict future performance. In KT, problems are labeled with needed skills (i.e. knowledge components, KC) [49], the skill or q-matrix can be learned from data [9], or the problem ID can be used instead. In Bayesian Knowledge Tracing (BKT), the most popular KT method [14], a simple Bayesian model is built to model student knowledge using parameters for guess (getting a problem right when a skill is not known), slip (getting it wrong when known), and transition from unlearned to learned after practicing. These parameters are learned from prior students’ problem sequences, and then used to predict future performance. Researchers have improved BKT performance, for example, by calculating the bound or prior distribution of parameters [8], adding a priori estimates of student learning [32], or integrating speed factors [56].

A number of innovations have improved *domain general* KT, without using additional features from student’s work (only the correctness of each problem attempt). With the development of machine learning technologies and increasingly available large datasets, models based on deep learning have been proven more effective, especially with enough available data [17]. Piech et al. introduced deep knowledge tracing (DKT), using recurrent neural networks (RNN) to predict a student’s knowledge of each skill (or problem) after each problem attempt, and to learn the relationships among skills automatically [34]. As our work is based on this model, we will discuss the details of the model in Section 3. Some recent advances in deep learning for knowledge tracing focus on model structure, including SAKT and SAINT. Self-attentive knowledge tracing (SAKT) [31] added a self-attention mechanism [54] to DKT, while Separated Self-Attentive Neural Knowledge Tracing (SAINT) [12] later integrated a transformer (a type of deep neural network which has been successfully applied in text and image processing areas) into a knowledge tracing model [51]. Both of these models have outperformed DKT, especially on large datasets such as EdNet [13], e.g. by 2% AUC.

While these innovations have improved KT performance, often using complex networks and larger datasets, the datasets used generally only indicate whether a student’s attempt was *correct*, but not the *content* of a student’s answer or their *process* for achieving it, and the models therefore do not use this information. However, researchers have incorporated *other types* of information into deep models, such as course

prerequisites or the relationships among problems. For instance, Chen et al. attached prerequisite information in the DKT modeling process for a more accurate prediction [10]. The prerequisite concepts were modeled as graph matrices (as done by Wang et al. [53]), serving as an additional input to knowledge tracing models, similar to skill or q-matrices that can also be learned from student data [9]. On the other hand, Ghosh et al. introduced attentive knowledge tracing (AKT) [18]. They introduced a decay parameter to explicitly reduce the impact of distant problems, and at the same time used a Rasch model [37] to incorporate problem contexts, then embedding the differences among the problems. Student information can also be used for knowledge tracing models. Educational priors such as a learning or forgetting curves can be integrated into deep knowledge tracing models [11]. The closest such models come to incorporating students’ solution processes is including information about how *fast* students solved a problem. Yudelson et al. [56] added speed factors into BKT, and similar temporal information can also improve the performance of deep models (e.g. [44]).

None of the above work have used the student response information, besides submission correctness, in their models. This could be partially because of the simplicity of the problems. Most of them are true or false, multiple choice problems, or short answer problems. The availability of the exercise data is also limited, as some datasets only contains a sequence of binary correctness scores from students. Recent work (e.g. EKT, EERNN [45, 26]) used a joint embedding of *exercise* text and response correctness, combining the exercise text embedding together with student scores to represent student individualized submissions. This achieved better performance than the other models without using this information. However, these models only use *problem information*, but no information about the *students’ answer* beyond binary correctness information. This suggests an opportunity to create improved, *domain-specific* KT models in areas such as programming, math, science or writing, where students’ answers include complex written responses or structured problem-solving steps. Recent work has incorporated such problem-specific data in deep learning approaches used to adapt pedagogical policies for tutoring in logic [27], probability [58], or predict performance in programming [29] but generally have not been used to built KT models in these domains. In the domain of programming education, for example, students’ code submissions contain rich information on the state of their current knowledge. As proposed in this paper, more structural information could be extracted from student code submission to infer students’ learning status of certain concepts. We use these code features to make better knowledge tracing models.

2.2 Student Modeling in CS Education

Researchers in CS education have explored ways to model student source code for intelligent tutoring. In 2011, Jin et al. proposed that a linkage representation that reflected code structure could be used for programming hint generation [23]. In 2014, Yudelson et al. extracted code features from a MOOC on introductory Java programming to explore code recommendation methods [55]. Their work focused on using a combination of problem correctness and extracted code features to predict student success, and use this prediction to recommend an appropriate next problem to a stu-

dent. While they did not evaluate their model on a KT task per se, their approach of extracting atomic code features is somewhat similar to our TFIDF baseline (Section 4.3). Another work from Rivers et al. used code features for student learning curve analysis and attempted to directly extract meaningful knowledge components (and whether they were successfully applied) from student code [39]. In their work, student code submissions are represented as abstract syntax trees (ASTs), with the node types of ASTs (e.g. `for`, `if`) representing knowledge components (KCs). The error rate curves (referred to as “learning curves”) were plotted over time, visualizing the mastery of different KCs. They showed that while code-based KCs produced well-fitting curves, others did not. While this suggests the possible validity of AST-based KC extraction, the work did not directly evaluate the utility of these KCs for knowledge tracing. Like our current work, Wang et al. showed that incorporating structural code features can improve DKT for a single problem from a large “hour of code” (HoC) dataset [52]. However, this HoC exercise has a very simple solution, so their results may not generalize. Additionally, their features were learned in an unsupervised way from ASTs, while our approach learns an embedding from the data.

Code features have also been used in tasks other than KT as well, such as common bug identification in student code. Traditionally, experts manually examined student code to identify common bugs in different student levels and programming languages, such as Java [48] or block based programs [20]. However, manual examination is expensive for large-scale and quantitative studies. More advanced work takes advantage of the growing size of datasets, and used data-driven methods to find bugs in student code submissions. For example, Choi et al. used simple machine learning methods to detect malicious code in code by using simple feature extraction methods such as counting neighboring tokens in code text (n-gram). With the recent advance of computational power and even bigger datasets, more deep learning methods have emerged. These methods focused on developing deep neural network methods to extract structural information for automatic student bug detection. For example, Gupta et al. used a matrix to represent the ASTs of student code to localize student code submissions [19] in a large dataset (270K samples). For smaller sized dataset, Shi et al. evaluated the bug detection performance with the help of semi-supervised learning [42], and have also shown that unsupervised learning is possible with the help of experts [43]. All these methods reported better performance than traditional data-driven models on their tasks, showing the feasibility of similar usage on KT tasks.

While we focus on using student code submissions to extract features for student programming KT tasks, other less complicated approaches exist. Original programming tutors such as ACT [15] and Lisp tutor [6] decompose computational problems into small steps and let students make choices. This facilitates the KT tasks, as in these datasets, student submissions are simple multiple choices. However, with the development of newer Intelligent Tutoring Systems (ITSs), more systems provide intelligent support to students’ written code. This provides better practice for students, but also makes knowledge tracing in computer science a more challenging task. Our paper aims at extracting code fea-

tures for KT tasks in these new datasets.

2.3 Deep Code Learning

Besides code feature extraction in the CS education domain, programming code has also been analyzed with data-driven models in software engineering research. For example, Allamanis et al. used neighboring tokens in source code (n-grams) to represent programming code, borrowing methods from natural language processing studies to predict method names in big code datasets [3]. Later work further explored extracting features from code structure, such as Raychev et al. who used decision trees to model programming code, making probabilistic predictions on the types of nodes in AST [38]. However, these simple structural approaches are often outperformed by newly developed deep learning models, especially when applied to big datasets.

Deep neural networks have been applied in the software engineering domain, and achieved better performance than traditional data-driven methods. For example, Allamanis et al. used convolutional neural networks (CNNs) to classify code functions [2]; Mou et al. reworked the CNNs to an AST version, using the parent-children direction information in tree representations. Both methods greatly improved method classification tasks on classical machine learning models. Another recent model, code2vec, outperformed these models. Alon et al. designed this model, which leverages nodes and traversal paths in the ASTs to represent programs [5]. In their work, the leaf nodes of the ASTs are selected to represent the semantic information about the code. In addition, as there is a path through the AST from every leaf node to any other leaf node, this path is extracted to represent the code’s structural information. The traversal paths together with the corresponding leaf nodes serve as the basic units of a representation of code [4]. The code2vec model calculates the weight of each code path using an attention mechanism [54] to automatically classify function names. Code-DKT’s code extraction component is based on the code2vec model, but adds score to the attention mechanism to assign weights to code paths [4] for predictions.

We chose code2vec to represent student code in DKT due to its recent successes for modeling code, and its attention mechanism. The attention mechanism learns weights for different features, allowing the model to directly use score information to select the most predictive code paths. Future work could investigate other code representations such as ASTNN, which has also been applied to make predictions from student code [28], or more recent advances such as CodeBERT [16].

3. METHOD

Problem Definition: Knowledge tracing (KT) tasks model a prediction problem: Given the history of a student’s attempts at various KCs/problems, the model predicts if the student will succeed on their next attempt ¹. Specifically, we define each student attempt \mathbf{x}_t at time t as (q_t, a_t, c_t) , where q_t is the problem ID, a_t is the correctness, and c_t is the program code submitted for this attempt. Historically, KT algorithms have only utilized q_t and a_t , and in this work we extend the input sequence to include c_t . At each timestep T ,

¹We use problemIDs for KCs in this work

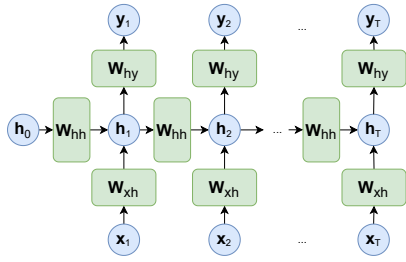


Figure 1: Recurrent neural network structure.

the model is given the T -length student attempt sequence $\mathcal{S}_T = \{(q_1, a_1, c_1), (q_2, a_2, c_2), \dots, (q_T, a_T, c_T)\}$, and it predicts whether the student’s next attempt ($T + 1$) on a given problem (q_{T+1}) will be correct (a_{T+1}). Note that students may attempt problems multiple times, and the model will make a prediction at each attempt.

Our proposed Deep Code Knowledge Tracing (Code-DKT) model integrates deep knowledge tracing (DKT) [34] with the code2vec classification algorithm [5]. In this section we introduce the DKT model and how we enhance it with code feature extraction and selection.

3.1 Deep Knowledge Tracing

Deep knowledge tracing uses a recurrent neural network (RNN) structure to learn the probability that a student will make a correct attempt on a subsequent problem. In the original implementation of DKT, the authors also implemented a version of DKT using a long short term memory (LSTM) model [21], which is widely perceived as an advancement over RNNs. For simplicity, we explain DKT using an RNN model; we performed DKT using both RNNs and LSTMs. In the experiments, the LSTM version yielded higher performance² (see performance comparison in Section 5.1.4). We chose DKT as our baseline model, to compare with and to extend, as it is a commonly used baseline in other more recent KT papers [31, 44]. Further, its LSTM structure makes it straightforward to extend with code features and to directly evaluate those features’ contributions. Some recent models have outperformed DKT, but only by about 2-4% AUC [31, 44], suggesting that DKT is still representative of modern deep KT models.

Model Input: For each student, DKT (RNN) takes as input a sequence $\mathcal{S} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T\}$ of T attempt vectors \mathbf{x}_t . With M problems, each attempt consisting of problem-correctness pair $\{q_t, a_t\}$ at time t , is one-hot encoded into a binary vector \mathbf{x}_t of size $2M$, where $x_{q_t+M(1-a_t)}$ is set to 1, and the other bits are set to 0. For example, with $M = 3$, for student success on problem 1, $q_t = 1, a_t = 1$, so $x_{1+3(1-1)} = 1$, so $\mathbf{x} = \{1, 0, 0, 0, 0, 0\}$, and failure on problem 1 $q_t = 1, a_t = 0$, so $x_{1+3(1-0)} = 1$, so x_4 is set to one, and \mathbf{x} is $\{0, 0, 0, 1, 0, 0\}$.

Model Structure: The RNN version of DKT maps each input sequence \mathcal{S}_T into an output sequence of predictions $\mathcal{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_T\}$ with a set of hidden states $\mathbf{h}_1, \mathbf{h}_2, \dots, \mathbf{h}_T$. More specifically, as illustrated in Figure 1, this process is

²See the appendix of [34] for the LSTM DKT equations.

defined as:

$$\mathbf{h}_t = \tanh(\mathbf{W}_{xh}\mathbf{x}_t + \mathbf{W}_{hh}\mathbf{h}_{t-1}),$$

$$\mathbf{y}_t = \sigma(\mathbf{W}_{hy}\mathbf{h}_t).$$

In the equations, element-wise operators $\tanh(\cdot)$ and $\sigma(\cdot)$ are activation functions of the network, introducing non-linearity to the network. The parameters learned in the network are \mathbf{W}_{xh} which transforms input \mathbf{x}_t into the hidden space, \mathbf{W}_{hh} which fuses the hidden state \mathbf{h}_{t-1} from the prior input with the current hidden state \mathbf{h}_t , and \mathbf{W}_{hy} which translates the hidden state h_t into an output. In both equations, the bias terms are omitted for simplicity, and the \mathbf{h}_0 is the initial hidden state, the zero-vector.

Model Output: The output sequence \mathcal{Y} contains prediction vectors \mathbf{y}_t , sized M . Every element of the vector represents the probability of the student making a correct submission on corresponding problems in their next attempt. Note that while the model makes predictions for each problem at each timestep t , only the value for the next attempted problem q_{t+1} is used during training and evaluation.

3.2 Deep Code Knowledge Tracing

We extend DKT into Deep *Code* Knowledge Tracing (Code-DKT), by using the code2vec [5] representation of student code attempts, c_t , along with problem and correctness information.

Code Representation: Abstract syntax trees (ASTs) are used to represent the hierarchical structure of code, for example with a node for a function (`method`) with children representing the function’s parameter (`input`) and body (`body`). AST leaf nodes often correspond to literal values or identifiers. Code-DKT extends the code2vec model for code classification, which encodes an AST using a set of *leaf-to-leaf paths* throughout the AST. For example, in Figure 2, a path from the leaf node `input` to the leaf node `"value"` (highlighted red in the example) consists of the nodes: `[input, method, body, String, "value"]`. Given an AST, code2vec extracts a set of leaf-to-leaf paths, as explained below.

Model Input: Since a deep learning model cannot operate directly on code paths, the Code-DKT must next convert this code-path representation of the AST into a binary vector. A student’s code submission c_t at time t is represented as $\{p_0, p_1, \dots, p_R\}$ where there are in total R randomly selected code paths in c_t . Every p_r has three components, namely the starting node of the code path \mathbf{s}_r , the textual representation of the full path \mathbf{o}_r , and the ending node \mathbf{q}_r , which are each one-hot encoded as binary vectors. For instance, for the example in Figure 2, \mathbf{s}_r is `input`, \mathbf{o}_r is a text string: `input|method|body|String|value`, and \mathbf{q}_r is `value`.

Model Structure: Rather than using a *static* vector representation of students’ code, Code-DKT *learns* an optimal embedding of student code. The detailed Code-DKT model structure is shown in Figure 3. This initial structure is drawn from code2vec. The nodes for each of R code paths in c_t (c_t has in total R paths), including starting and ending nodes ($\mathbf{s}_r, \mathbf{q}_r$) and paths \mathbf{o}_r for a single path r , are respectively embedded by the node embedding matrix \mathbf{W}_{encode}

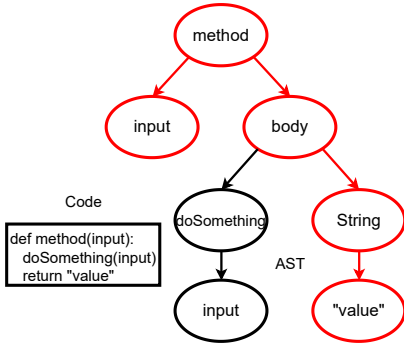


Figure 2: A simple AST where red nodes and edges represent a leaf-to-leaf path from input to "value".

and the path embedding matrix \mathbf{W}_{epath} . Both matrices are randomly initialized with a Gaussian distribution, but they are later updated during model training. The Code-DKT model structure then diverges somewhat from code2vec, to account for the specific needs of the KT problem. Specifically, the three embedded vectors representing c_t are concatenated with the problem-correctness vector \mathbf{x}_t from DKT (introduced in Section 3.1). This serves as a numerical representation of (q_t, a_t, c_t) . For a single code path p_r , this process is accomplished with embeddings for the start node ($\mathbf{e}_{s,r}$), path ($\mathbf{e}_{o,r}$), and end node ($\mathbf{e}_{q,r}$):

$$\mathbf{e}_{s,r} = \mathbf{W}_{enode} \mathbf{s}_r; \mathbf{e}_{o,r} = \mathbf{W}_{epath} \mathbf{o}_r; \mathbf{e}_{q,r} = \mathbf{W}_{enode} \mathbf{q}_r,$$

$$\mathbf{e}_r = [\mathbf{e}_{s,r}; \mathbf{e}_{o,r}; \mathbf{e}_{q,r}; \mathbf{x}_t].$$

Score-Attended Path Selection: Code-DKT now has an numerical representation of a single attempt: a set of R embedded vectors, \mathbf{e}_r , one for each code path in c_t . Note that the embedding, \mathbf{e}_r not only includes the code information, but also the current correctness score information \mathbf{x}_t at the submission t . However, not all parts of a student's code are relevant, and thus not all code paths \mathbf{e}_r are important for predicting a student's future success. Therefore, the model uses an attention mechanism to identify how much weight to give to each of these paths. The embedding vectors $\mathbf{E} = \{\mathbf{e}_0, \mathbf{e}_1, \dots, \mathbf{e}_R\}$ are multiplied by the attention matrix \mathbf{W}_a to get R scalars a_0, a_1, \dots, a_R , representing the importance (commonly known as the "attention") of each of the code paths. The importance a_r uses a SoftMax mechanism for normalization, having 1 as the sum. This process is formulated as:

$$\alpha = \text{SoftMax}(\mathbf{E}\mathbf{W}_a)$$

$$\text{SoftMax}(\mathbf{a}) = \frac{e^{a_i}}{\sum_{i=1}^R e^{a_i}}$$

where each elements α_r in $\alpha = \{\alpha_1, \alpha_2, \dots, \alpha_R\}$ are the calculated weights for the code path p_r . Finally, Code-DKT weights each code path \mathbf{e}_i by its attention α_i , and sums them together, giving a weighted average: a single vector representing the important parts of the code. The weighted average vector is then multiplied by a matrix \mathbf{W}_0 to get the code vector \mathbf{z} , representing features extracted from code

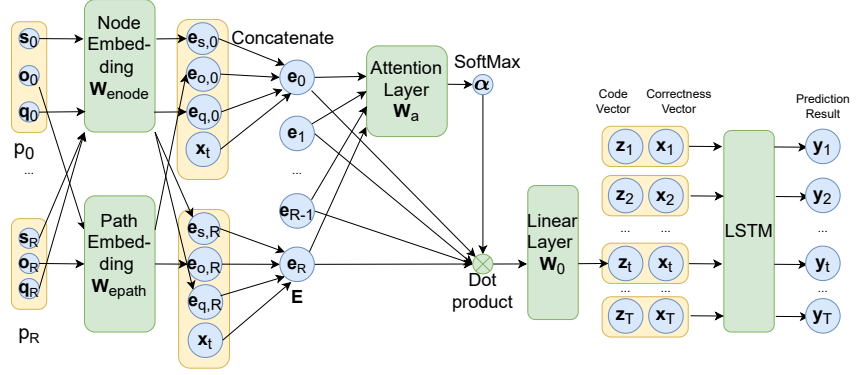


Figure 3: Code-DKT model structure.

submissions, as in equation:

$$\mathbf{z} = \mathbf{W}_0 \left(\sum_{i=1}^R \alpha_i \mathbf{e}_i \right).$$

In a sequence of T student attempts, Code-DKT produces T code vectors $\{\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_T\}$. The code vectors are concatenated with the correctness vectors $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T\}$ as the input to the final LSTM (as in DKT), giving the predictions $\{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_T\}$. Even though \mathbf{x}_t was already used to produce \mathbf{z}_t , this final concatenation ensures the Code-DKT model has direct access to the student correctness score information.

4. EXPERIMENTS

We designed an experiment to evaluate 3 research questions about student modeling in the domain of programming:

- RQ1** How effective are domain general KT approaches (DKT, BKT) on our programming dataset?
- RQ2** How can features derived from students' code be used to improve KT models?
- RQ3** When are these code features most useful, and how can they lead to improved predictions?

4.1 Dataset & Experiments Setup

Our study uses a dataset of an introductory Java programming class at a large, university in the US, collected in Spring 2019, stored in the ProgSnap2 format [36]. The dataset includes work from 410 students on 50 problems divided over 5 assignments. These were completed throughout the semester as homework, with each assignment focusing on a specific topic (e.g. conditionals, loops). For these problems, typical solutions ranged 10 to 20 lines of code. Students tended to make multiple submissions before succeeding finally, and 23.68% of the attempts were correct. Student code was automatically graded using test cases, and We treated a submission as correct (1) only when all test cases passed, and incorrect (0) otherwise.

For each assignment, students were then split into training and testing sets with a ratio of 4 : 1. One quarter of the training data were used for hyperparameter tuning and validation (see below). Then, we trained the model on the whole

Table 1: Performance Comparison on all assignments.

Model	A1	A2	A3	A4	A5
DKT	71.24%	73.09%	76.84%	69.16%	75.14%
Code-DKT	74.31%	76.56%	80.40%	72.75%	79.14%

Table 2: Overall and the first attempt performance of all models on assignment A1.

Models AUC (STD)	Overall	First Attempts
Code-DKT	74.31% (0.90%)	75.74% (0.69%)
DKT-TFIDF	69.94% (0.88%)	72.77% (0.79%)
DKT-Expert	69.52% (0.68%)	69.53% (0.72%)
DKT	71.24% (2.54%)	72.26% (3.69%)
BKT	63.78% (4.68%)	50.22% (2.86%)

training dataset, and tested on the holdout test dataset, repeating this process 10 times to account for model variation (e.g. due to random initialization). All deep learning models were implemented using the PyTorch[33] library, and our BKT implementation was pyBKT [7].

4.2 Hyperparameter Tuning & Optimization

For hyperparameter tuning, we split the training data into training and validation sets, and created a model with each possible set of hyperparameters (described below), and calculated AUC performance on the validation dataset. We repeated this process 100 times and chose the hyperparameter setting with the best average validation performance to use in testing/evaluation. Specifically, we selected the embedding size of code feature extraction as 300, from a range of (50, 100, 150, 300, and 350); learning rate was selected as 0.0005 from a range of (0.00005, 0.0005, 0.005, 0.01); the training epochs were set at 40 to save training time while keeping the best prediction results, selected from a range of (20, 40, 100). All other parameters were defaulted as the original settings of code2vec and DKT. We fixed the longest length of student attempts at 50 to filter extra long submission traces from students. In cases where more than 50 attempts were submitted, we used the last 50 submissions, assuming the latest submissions were more useful.

As the models were deep neural networks, we used binary cross entropy as a loss function to track the difference between the ground truth and predicted probabilities. The models used back propagation to update weight matrices (parameters), using the Adam optimizer [24], which is also a default for code2vec and DKT.³

4.3 Baselines

We compare the performance of Code-DKT to DKT, BKT, and two modified DKT methods: DKT-TFIDF adding data-driven features, and DKT-Expert adding expert features. Specifically, DKT-TFIDF uses TFIDF, a data-driven feature that counts the term frequency (TF) of tokens (variables, functions, and operations, etc.) in code text, and forms a frequency vector for every term. This frequency is multiplied by the inverse document frequency (IDF) to show how often terms show up in *unique* documents. As students use various

variable names, we limited the top 50 best features (selected from a range of (30, 50, 100, 300) in hyperparameter tuning) in TFIDF to remove redundant features. For the DKT-Expert model, two authors examined the problems in the dataset, and determined 9 rule-based code features. These features include code component existence checks such as the usage of `else if` statements, the usage of `&&` operations, etc. These statements and operations represent students’ usage of certain concepts such as writing alternative conditions, or using “and” logic to solve a problem.

To improve the TFIDF and Expert models to serve as more robust baseline models, we added one additional set of features (only to baseline models) to encode information about the skills practiced in each problem, as has been done in prior work [57]. Two authors examined the problem descriptions and solutions and agreed on 9 skills we expected students to learn. For example, one skill was solving problems with negative conditions in the instructions (using words such as “unless”, “otherwise”), requiring students to negate these conditions in their code. We represented each problem as a binary vector of practiced skills, and we used this skill vector to represent problems, instead of the one-hot encoded problem ID (see Section 3.1 for model input encoding). Testing on the validation dataset showed slightly improved performance using these skill vectors.

Metric: Our primary performance metric is AUC, a standard evaluation metric for KT models [34, 44, 31], as it uses the predicted *probability* of success, rather than a binary correctness prediction, and is more appropriate than accuracy for imbalanced datasets like ours (23% positive).

5. RESULTS

5.1 Performance Comparison

5.1.1 Code-DKT vs DKT

Table 1 shows a comparison of DKT and Code-DKT across all 5 assignments (the average of the 10 test runs). Note that for each assignment, a new model is trained and tested separately, without using data from prior assignments. This was because assignments were spaced out with weeks between them, including additional learning content, so students’ performance on prior assignments is less relevant. To address RQ1, we consider the overall performance of the baseline DKT model on our dataset, which has an AUC of 69-75% across assignments. This low score means it may be difficult to use model predictions to inform instruction or an automated intervention, as we discuss in Section 6. To address RQ2, we see that Code-DKT *consistently* outperforms DKT by 3-4% AUC on each assignment. This shows that our approach, which augments correctness features with additional information from student code, can improve DKT predictions. For perspective, this improvement is comparable to SAINT+’s improvement over DKT on EdNet (+2.76%) [13], or SAKT’s improvement on various datasets (+3.8%) .

5.1.2 Code-DKT vs Naive Code Features and BKT

We now investigate a single assignment, A1, to illustrate Code-DKT’s performance, and create a DKT-Expert baseline using assignment-specific, expert-authored code features. We selected assignment A1, as it came first (and was therefore not influenced by prior assignments) and its skills are

³Repository: <https://github.com/YangAzure/Code-DKT>

the least complex. Table 2 shows the performance of Code-DKT, DKT, as well as 3 new baselines: BKT, and 2 simple code-feature extensions of DKT: DKT-TFIDF and DKT-Expert (described in Section 4.3). Model performance is given for predicting all attempts (Overall) and for predicting only first attempts at each problem. The results show that neither the simple expert features nor the TFIDF data-driven features improve the overall performance of DKT. These simple features derived from student code instead negatively affect overall performance. This suggests that a more effective model structure is necessary for making use of code features, such as our Code-DKT model. We also see that BKT has an AUC score of only 63%, suggesting that deep models are more effective for our dataset.

5.1.3 When is Code-DKT Effective?

We used assignment A1 to investigate *when* Code-DKT was more effective than DKT, helping to answer RQ3.

Overall vs First Attempts: We investigated Code-DKT’s performance at predicting a student’s *first* attempt at each problem (Table 2, column 3). First attempts are important in a KT task because they represent points at which an ITS might make key interventions (e.g. offering a worked example if a student might fail at problem solving). Therefore, many KT evaluations differentiate a student’s first attempt on a task (where a model must make predictions using only performance on *other* problems) from subsequent attempts. This distinction also helps us understand when the Code-DKT model is most effective. One might ask, is Code-DKT using student code submissions to learn a better representation of student knowledge (which *would* help it predict first attempts), or is it simply estimating how close a student is to solving the *current* problem (which would only help to predict *subsequent* attempts). Our results shows that Code-DKT actually performs *best* when predicting first attempts, and it also shows a similar improvement over DKT for first attempts (+3.48%), compared to all attempts (+2.93%) . This suggests that the content of a student’s code is helpful for not only predicting how quickly they will solve the *current* problem, but also *future* problems.

Problem-specific Performance: Table 3 shows the decomposed AUC performance of Code-DKT and DKT on each problem. We observe that Code-DKT outperforms DKT overall on 6 of the 9 problems. The difference ranges from +15.54% AUC (problem 13) to -4.43% (problem 236), suggesting that the benefit of Code-DKT’s code features depends somewhat on the programming problem. It also shows that code features *can* reduce model performance, but the potential for Code-DKT’s improvement seems to be greater than the potential for harm.

To understand *when* Code-DKT’s code features were useful, we investigated differences between the problems where it outperformed DKT and those where it did not. We found that many of the problems where there was improvement shared similar learning concepts and solution structure. For example, problems 3, 232 and 234 all used the “independent choice” programming pattern, which is often solved with nested if-statements. Similarly, problems 1, 3, 5 and 13 all included a pattern where one condition changes a value used in another condition. These common patterns seem to have

Table 3: Decomposed performance of Code-DKT and DKT AUC performance on different problems in assignment A1.

Problems	Code-DKT		DKT	
	Overall	First	Overall	First
234	64.60%	71.38%	63.75%	73.48%
13	78.45%	86.55%	63.59%	68.81%
232	74.93%	78.99%	72.49%	73.09%
233	64.79%	74.57%	67.18%	76.33%
5	75.38%	81.34%	74.28%	81.79%
235	70.65%	71.96%	75.03%	70.80%
236	74.25%	74.30%	78.68%	77.06%
1	68.62%	70.32%	66.67%	73.20%
3	71.00%	71.00%	64.02%	64.02%

helped the model make better predictions on problems that used them. However, 2 of the 3 of the problems where Code-DKT performed poorly involved a unique learning concept that did not appear in any other problems. For example, problem 236 requires students to check if any 2 of the 3 given variables are equal (which has no analog among other problems) and 233 requires the `Math.abs` function (which many students failed to use correctly). Together, these results suggest a hypothesis that Code-DKT’s code features are most useful at predicting problems that share code structures with other problems, and less useful at predicting problems that emphasize novel code structures. This suggests Code-DKT may be successfully modeling students’ knowledge of common code patterns.

5.1.4 Ablation Study

Our Code-DKT model design choices include: where to incorporate correctness information, how to update the embedding, and what underlying network to use (LSTM or RNN). Table 4 shows the results of an ablation study on assignment A1 to determine which of these choices improved the performance of our final DKT model (first row). The final Code-DKT model concatenates the *correctness* of a students’ attempt with code features in two places (see Section 3.2): before the attention mechanism (the vector \mathbf{e}_r), and in the final trace fed into the LSTM (\mathbf{z}_i concatenated with \mathbf{x}_i). The model in row 2 only includes correctness information in the first case, and row 3 includes it only in the second case. Both models lose performance, but not by much (0.5%), suggesting that correctness information helps both in attending to relevant code paths, and final predictions, but this information is somewhat redundant. We also investigated using an RNN (row 4) instead of an LSTM, but this was, as predicted, moderately less effective. Finally, recall that Code-DKT uses `code2vec` to embed students’ code as a vector, and updates this embedding throughout model training. Row 5 shows a version where we pretrained this embedding on the training dataset, using `code2vec` to predict the correctness of students’ code, and then fixed the embedding when training the LSTM. This model does much worse, suggesting that the relevant features for predicting the *correctness* of code are different from those for predicting *future performance*.

5.2 Case Studies

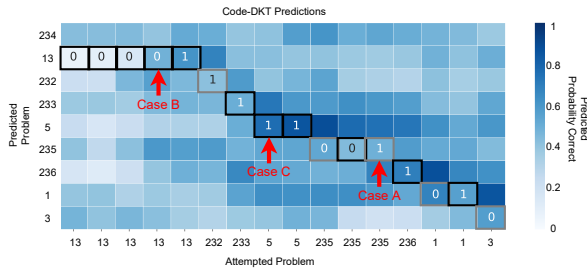


Figure 4: Code-DKT generated correctness predictions heatmap for a student.

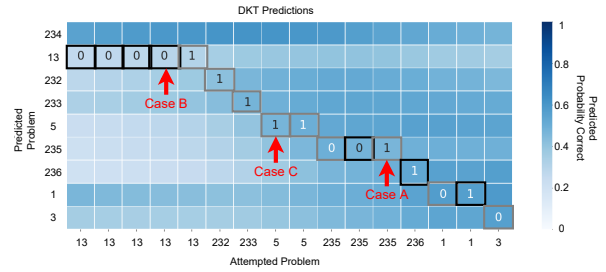


Figure 5: DKT generated correctness predictions heatmap for a student.

Table 4: Code-DKT ablation study on A1.

	Model	Overall AUC
1	Code-DKT (Final Model)	74.31%
2	Correctness: Attention Only	73.81%
3	Correctness: Trace Only	73.84%
4	Model: RNN	73.63%
5	Embedding: Static	68.74%

To further answer RQ3, we examined *how* code features may have improved Code-DKT through 3 case studies. We use prediction heatmaps from Code-DKT and DKT for one student, shown in Figures 4 and 5 for Code-DKT and DKT, respectively. The rectangular cells show which problem the student actually attempted (y-axis) at each time-step (x-axis), and the numbers in the cells represent the ground truth values of whether student’s attempt was successful (1) or unsuccessful (0). Black frames indicate correct (i.e. accurate) model predictions, while grey ones indicate incorrect predictions. The color of the heatmap in each cell specifies the predicted probability of students making a correct submission on a given problem (y-axis) at the given time-step (x-axis), and darker means a higher probability of success. For example, in Figure 4, the student makes 4 unsuccessful attempts at problem 13, followed by a successful attempt, then succeeds at problems 232 and 233 in one attempt each.

The heatmaps for the student (Figures 4 and 5) show that Code-DKT is able to make better predictions on the traces than DKT, making 11 out of 16 successful predictions, while DKT is able to make 8 of them correct. Another observation is that Code-DKT heatmaps have much stronger predictions with values close to 1 or 0 compared with DKT, showing that with code features, the model is more confident.

Case A: Successful Prediction: In Case A, Code-DKT uses code features to make better predictions than DKT on the predictions of the student’s final submission on Problem 235. As shown in Figures 4 and 5, while both Code-DKT and DKT can successfully predict the incorrect submission on the student’s second submission of Problem 235 and fail to predict the correct submission on the third, Code-DKT gives a higher prediction than DKT. In Figure 6, the student’s code submissions show the reason. The student’s second submission is almost correct, demonstrating a correct (if inefficient) nested if-else structure, but they have omitted the nested condition in their else branch. Code-DKT is able to

```

public int dateFashion(int you, int date)
{
    int value = 0;
    if (you >= 8 || date >= 8)
    {
        if (you <= 2 || date <= 2)
        {
            value = 0;
        }
        else
        {
            return 2;
        }
    }
    else
    {
        value = 1;
    }
    return value;
}

public int dateFashion(int you, int date)
{
    int value = 0;
    if (you >= 8 || date >= 8)
    {
        if (you <= 2 || date <= 2)
        {
            value = 0;
        }
        else
        {
            return 2;
        }
    }
    else
    {
        if (you <= 2 || date <= 2)
        {
            value = 0;
        }
        else
        {
            value = 1;
        }
    }
    return value;
}

```

Figure 6: Code at times t and $t + 1$ for Case A, where the code c_1, \dots, c_t is used to predict correctness at $t + 1$.

infer the quality of the student’s code, since its prediction of success probability increased from 44.2% to 49.1% after the student’s second (incorrect) attempt, while DKT’s prediction decreased from 47.7% to 46.7%. Code-DKT’s higher prediction may be because the if-else structure the student was missing was very similar to one they had already written, as shown in Figure 6. These code structures are easily captured by the path-based AST representation used by code2vec. Without code features, it is difficult for DKT to predict whether the student is going to succeed on $t + 1$, since it only knows the student has failed twice, not how close they are to succeeding. Even with code features, there is still a great deal of uncertainty. No matter how close a student is to a correct answer, there is no guarantee they will achieve it on their next attempt. This may help to explain why Code-DKT does not more dramatically outperform DKT overall.

Case B: Unsuccessful Prediction: Case B shows that even when a student’s code is nearly correct for a given problem, it doesn’t guarantee that they will be successful on their next attempt. Sometimes Code-DKT is overconfident in these situations, and incorrectly predicts success, as in Case B. Figure 7 shows the last three attempts the student made on Problem 13: two incorrect followed by a final correct attempt. The only differences between the final attempt and the earlier two is shown in the red frames. The student’s 4th attempt achieved the correct logic for Problem 13, the


```

else
{
  if (
  {
    else
    {
      if (speed <= 60)
      {
        ret
        {
          if (speed <= 60)
          {
            value = 0;
          }
          if (speed >= 61 && speed <= 80)
          {
            value = 1;
          }
          if (speed >= 81)
          {
            value = 2;
          }
        }
      }
    }
  }
}
return;
}
return value;

```

Figure 7: Case B 4th, 5th and 6th code attempts.

```

public String alarmClock(int day, boolean vacation)
{
  String time = "";
  if (vacation)
  {
    if (day <= 5 && day >= 1)
    {
      time = "10:00";
    }
    if (day == 0 || day == 6)
    {
      time = "off";
    }
  }
  else
  {
    if (day <= 5 && day >= 1)
    {
      time = "7:00";
    }
    if (day == 0 || day == 6)
    {
      time = "10:00";
    }
  }
  return time;
}

public boolean love6(int a, int b)
{
  if (a == 6 || b == 6)
  {
    return true;
  }
  return (a + b) == 6 || Math.abs(a - b) == 6;
}

public boolean answerCell(boolean isMorning, boolean isMom, boolean isAsleep)
{
  if (isAsleep)
  {
    return false;
  }
  if (isMom)
  {
    return true;
  }
  return !isMorning;
}

```

Figure 8: Case C, using code from problems 232 and 233 to predict the same student’s performance on problem 5

5th attempt adds an empty `return` statement, and the 6th and final attempt adds the appropriate return value. After seeing the almost-correct code at their 4th attempt, Code-DKT predicted that the student would succeed on the next attempt since the modifications they needed were minimal (just write “`return value;`”), but it took one extra attempt to get it right. An expert might make a similar conclusion, that the student was close enough to realize their mistake and submit a correct answer, and would have similarly been wrong. This highlights the uncertainty present in any KT task and the challenges of applying KT to student code.

Case C: Successful Prediction (First Attempt): Case C illustrates that Code-DKT can also use code from *previous problems* to improve its predictions of *first attempts* of new problems (as shown quantitatively in Table 2). For example, when the student successfully completes problems 232 and 233 in a single attempt, Code-DKT’s prediction of the student’s success on problem 5 increases from 42.0% to 50.0% to 72.5% respectively, leading it to successfully predict success on the student’s first attempt at problem 5. However, DKT’s confidence only modestly increased from 41.8% to 47.3% to 47.0%, leading it to incorrectly predict failure. Both models know that these problems are related, and share some learning concepts (based on how other students’ successes on the problems are related), but Code-DKT’s analysis of the student code allowed it to infer more about the knowledge that

was demonstrated in past problems.

We use these three consecutive code submissions to explain why this may be the case in Figure 8. For example, in Problem 232, the student directly uses Boolean variable in the if-condition (`if (vacation)`) rather than a superfluous comparison (`if (vacation == true)`) that many students use, demonstrating a higher level of understanding. This same direct usage of Boolean variables is seen in the if condition and `return` statement of Problem 5. The code submission on Problem 233 further suggests the student is able to combine logical operators with Boolean variables to `return` a Boolean expression. This occurs again in the `return` statement of the students’ attempt at Problem 5, shown in the lower rectangular. While we cannot know for certain which code features Code-DKT used to make its success prediction for Problem 5, these repeated code structures are one possibility, given code2vec’s ability to recognize repeated patterns in ASTs.

6. DISCUSSION

RQ1: How well do domain-general models perform? We used domain-general KT models (DKT) as the baseline models for our programming dataset. These models performed relatively poorly, averaging 73.09% AUC across assignments. While this is considerably better than chance, the performance may not be high enough to use in some student modeling contexts. For example, for assignment A1, the recall of DKT was 31.4% and the precision was 46.5%, so the model fails to identify two thirds of unsuccessful attempts, and over half of the time when the model predicts a failed attempt, the student actually succeeded. This suggests that KT is a difficult challenge on this dataset. By contrast, DKT has historically been effective on other datasets, which are both larger and in other domains, such as EdNet [13], Assistments [41] and KhanAcademy [34]. One possibility is that the more complex nature of programming problems, with myriad possible correct and incorrect solutions, makes KT prediction more challenging on this dataset, compared to those in other domains. If this is the case, several aspects of programming may contribute to the challenge of modeling student success. Programming problems often require many attempts to get correct (6.1 on average in our dataset), leading to class imbalance. In our dataset, the problem descriptions were complex, and their solutions involved complex conditional logic, and students had to write perfect Java syntax for the program to compile. These factors mean there are many ways for students to make small “slips”, making the relationship between skill and success less direct.

Another possibility is that our dataset (410 students) was simply too small for complex deep models to find success, compared to the 1000s or even 100,000s of learners in other datasets where DKT has been evaluated. However, model complexity alone does not explain the difference, since the simpler BKT model did even worse than DKT, and our Code-DKT model, which had far more parameters, performed better. Additionally, DKT has historically performed well on some other small datasets (e.g. the “ASSIST-Chall” and “STATICS” datasets from [31] with 300-700 students). Regardless, many tutoring systems only have hundreds of students, and effective KT models must still be able to perform well on these small datasets. Thus, to the extent that

our datasets is representative of the domain, our results suggest the need for improved KT models for programming.

RQ2: How can code features improve KT models? Our results show that a simple extension of DKT with code features does not improve its performance. This result is somewhat surprising, given that relatively simple features (e.g. the presence of a `return` statement) should be at least somewhat related to how close a student is to a correct answer. It is possible such features may improve a model with different structure, but in our dataset, they were not helpful to DKT. This suggests the need for thoughtful approaches to incorporating domain-specific features into deep models. Our Code-DKT model was able to make reasonable improvements to DKT (+3.07% overall on A1). This is comparable to the improvement of SAINT over DKT on the EdNet dataset (they achieved +2.76% in AUC), or SAKT over DKT on various datasets (+3.8%) [31]. This suggests that domain-specific features can be just as important as model structure for effective KT. Code-DKT’s improvement is also robust. It has a +3% to +4% improvement overall on all five assignments. Importantly, however this is still a relatively poor performance overall, suggesting the need for more work on leveraging domain-specific features for improved KT.

RQ3: When and how do code features work? We also explored when and how the code features improved model performance. We found that code features are most useful on problems that share similar learning concepts with other problems in the dataset, and less useful on problems with unique and difficult concepts (e.g. `Math.abs()`). This makes sense – if we make an analogy to the original BKT where each problem was labeled with KCs, if you had a unique KC, the model would have no way of predicting on that problem. In our case, the KCs are inferred by the model, but the same limitation exists. However, most problems in our dataset did share primary learning concepts (e.g. loops, conditionals) and benefit from code features, and this repeated practice is a common feature of many CS1 courses. We also found that code features are useful for predicting both first attempts and subsequent attempts. Our case studies reveal potential mechanisms for both of these effects. For repeated attempts, the model seems to use the relative correctness of a student’s code to determine how close they are to a solution and therefore how likely they are to get it right on the next attempt. For first attempts, the model seems to identify code structures in prior attempts that indicate knowledge or competence with certain programming concepts, which it uses to make predictions on new problems. More work is needed to verify these hypotheses, and to understand how the model represents this knowledge.

Limitations: Our model and experiment have several limitations. 1) All models evaluated, including Code-DKT, have a relatively low performance, partially due to the difficulty of the problem and low data size (410 students), as discussed in Section 6. Still, they perform considerably better than chance, and such models could still be useful, e.g. in prioritizing help to struggling students. 2) Our dataset was from a single semester of a course. While our semester-long dataset of 50 problems is considerably more robust than some of the prior work on KT in programming (e.g. using 1-2 problems [52]), it is unclear how our results will gener-

alize to other semesters, classes or programming languages. 3) We used only DKT as a baseline model to extend and to compare against, and it is possible code features may have different effects on other models. However, as explained in Section 3.1, DKT has a comparable performance to more modern deep models, and made sense as a starting point to explore the effect of code features.

7. CONCLUSION

The contributions of the paper are 1) the Code-DKT model, which extends DKT with embedded code feature extraction; 2) results showing that CodeDKT consistently improves over DKT in a programming dataset; and 3) comparisons and case studies highlighting when and why Code-DKT code features help. This paper compared our new Code-DKT model to domain-general BKT and DKT baselines, and two DKT models extended with simple code features, demonstrating improved performance for Code-DKT over these baselines. However, the best baseline model performance was about 73%, and Code-DKT was 74.3%, demonstrating considerable room for improvement on modeling for knowledge tracing in programming. The case studies in this paper illustrate specific situations where knowledge tracing can be particularly difficult in programming, and where there is potential for improving code KT, e.g. when common code structures are used across problems.

Acknowledgements: This material is based upon work supported by NSF under Grant No. #2013502.

8. REFERENCES

- [1] F. Ai, Y. Chen, Y. Guo, Y. Zhao, Z. Wang, G. Fu, and G. Wang. Concept-aware deep knowledge tracing and exercise recommendation in an online learning system. 2019.
- [2] M. Allamanis, H. Peng, and C. Sutton. A convolutional attention network for extreme summarization of source code. In *International Conference on Machine Learning*, pages 2091–2100. PMLR, 2016.
- [3] M. Allamanis and C. Sutton. Mining source code repositories at massive scale using language modeling. In *2013 10th Working Conference on Mining Software Repositories (MSR)*, pages 207–216. IEEE, 2013.
- [4] U. Alon, M. Zilberstein, O. Levy, and E. Yahav. A general path-based representation for predicting program properties. In *Proceedings of the 39th ACM SIGPLAN Conference on Programming Language Design and Implementation*, pages 404–419, 2018.
- [5] U. Alon, M. Zilberstein, O. Levy, and E. Yahav. code2vec: Learning distributed representations of code. *Proceedings of the ACM on Programming Languages*, 3(POPL):1–29, 2019.
- [6] J. R. Anderson and B. J. Reiser. The lisp tutor. *Byte*, 10(4):159–175, 1985.
- [7] A. Badrinath, F. Wang, and Z. Pardos. pybkt: An accessible python library of bayesian knowledge tracing models. In *In: Proceedings of the 14th International Conference on Educational Data Mining (EDM 2021)*. ERIC, 2021.
- [8] R. S. Baker, A. T. Corbett, and V. Aleven. More accurate student modeling through contextual

- estimation of slip and guess probabilities in bayesian knowledge tracing. In *International conference on Intelligent Tutoring Systems*, pages 406–415. Springer, 2008.
- [9] T. Barnes. The q-matrix method: Mining student response data for knowledge. In *American Association for Artificial Intelligence 2005 Educational Data Mining Workshop*, pages 1–8. AAAI Press, Pittsburgh, PA, USA, 2005.
- [10] P. Chen, Y. Lu, V. W. Zheng, and Y. Pian. Prerequisite-driven deep knowledge tracing. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 39–48. IEEE, 2018.
- [11] Y. Chen, Q. Liu, Z. Huang, L. Wu, E. Chen, R. Wu, Y. Su, and G. Hu. Tracking knowledge proficiency of students with educational priors. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 989–998, 2017.
- [12] Y. Choi, Y. Lee, J. Cho, J. Baek, B. Kim, Y. Cha, D. Shin, C. Bae, and J. Heo. Towards an appropriate query, key, and value computation for knowledge tracing. In *Proceedings of the Seventh ACM Conference on Learning @ Scale*, pages 341–344, 2020.
- [13] Y. Choi, Y. Lee, D. Shin, J. Cho, S. Park, S. Lee, J. Baek, C. Bae, B. Kim, and J. Heo. Ednet: A large-scale hierarchical dataset in education. In *International Conference on Artificial Intelligence in Education*, pages 69–73. Springer, 2020.
- [14] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-adapted Interaction*, 4(4):253–278, 1994.
- [15] A. T. Corbett and A. Bhatnagar. Student modeling in the act programming tutor: Adjusting a procedural learning model with declarative knowledge. In *User Modeling*, pages 243–254. Springer, 1997.
- [16] Z. Feng, D. Guo, D. Tang, N. Duan, X. Feng, M. Gong, L. Shou, B. Qin, T. Liu, D. Jiang, et al. Codebert: A pre-trained model for programming and natural languages. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1536–1547, 2020.
- [17] T. Gervet, K. Koedinger, J. Schneider, T. Mitchell, et al. When is deep learning the best approach to knowledge tracing? *Journal of Educational Data Mining*, 12(3):31–54, 2020.
- [18] A. Ghosh, N. Heffernan, and A. S. Lan. Context-aware attentive knowledge tracing. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2330–2339, 2020.
- [19] R. Gupta, A. Kanade, and S. Shevade. Neural attribution for semantic bug-localization in student programs. *Advances in Neural Information Processing Systems*, 32, 2019.
- [20] F. Hermans and E. Aivaloglou. Do code smells hamper novice programming? a controlled experiment on scratch programs. In *2016 IEEE 24th International Conference on Program Comprehension (ICPC)*, pages 1–10. IEEE, 2016.
- [21] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.
- [22] B. P. Iii, A. Hicks, and T. Barnes. Generating hints for programming problems using intermediate output. In *In: Proceedings of the 7th International Conference on Educational Data Mining (EDM 2014)*, 2014.
- [23] W. Jin, L. Lehmann, M. Johnson, M. Eagle, B. Mostafavi, T. Barnes, and J. Stamper. Towards automatic hint generation for a data-driven novice programming tutor. In *Workshop on Knowledge Discovery in Educational Data, 17th ACM Conference on Knowledge Discovery and Data Mining*, 2011.
- [24] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015*, 2015.
- [25] K. Leelawong and G. Biswas. Designing learning by teaching agents: The betty’s brain system. *International Journal of Artificial Intelligence in Education*, 18(3):181–208, 2008.
- [26] Q. Liu, Z. Huang, Y. Yin, E. Chen, H. Xiong, Y. Su, and G. Hu. Ekt: Exercise-aware knowledge tracing for student performance prediction. *IEEE Transactions on Knowledge and Data Engineering*, 33(1):100–115, 2019.
- [27] M. Maniktala, C. Cody, A. Isvik, N. Lytle, M. Chi, and T. Barnes. Extending the hint factory for the assistance dilemma: A novel, data-driven helpneed predictor for proactive problem-solving help. *Journal of Educational Data Mining*, 12(4):24–65, Dec 2020.
- [28] Y. Mao, Y. Shi, S. Marwan, T. W. Price, T. Barnes, and M. Chi. Knowing” when” and” where”: Temporal-astnn for student learning progression in novice programming tasks. In *In: Proceedings of the 14th International Conference on Educational Data Mining (EDM 2021)*, 2021.
- [29] Y. Mao, R. Zhi, F. Khoshnevisan, T. W. Price, T. Barnes, and M. Chi. One minute is enough: Early prediction of student success and event-level difficulty during a novice programming task. In *In: Proceedings of the 12th International Conference on Educational Data Mining (EDM 2019)*, 2019.
- [30] B. Mostafavi and T. Barnes. Evolution of an intelligent deductive logic tutor using data-driven elements. *International Journal of Artificial Intelligence in Education*, 27(1):5–36, 2017.
- [31] S. Pandey and G. Karypis. A self-attentive model for knowledge tracing. In *In Proceedings of the 12th International Conference on Educational Data Mining (EDM) 2019*, 2019.
- [32] Z. A. Pardos and N. T. Heffernan. Modeling individualization in a bayesian networks implementation of knowledge tracing. In *International Conference on User Modeling, Adaptation, and Personalization*, pages 255–266. Springer, 2010.
- [33] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

- [34] C. Piech, J. Bassen, J. Huang, S. Ganguli, M. Sahami, L. J. Guibas, and J. Sohl-Dickstein. Deep knowledge tracing. *Advances in Neural Information Processing Systems*, 28, 2015.
- [35] T. W. Price, Y. Dong, and D. Lipovac. isnap: towards intelligent tutoring in novice programming environments. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*, pages 483–488, 2017.
- [36] T. W. Price, D. Hovemeyer, K. Rivers, G. Gao, A. C. Bart, A. M. Kazerouni, B. A. Becker, A. Petersen, L. Gusukuma, S. H. Edwards, et al. Progsnap2: A flexible format for programming process data. In *Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education*, pages 356–362, 2020.
- [37] G. Rasch. *Probabilistic models for some intelligence and attainment tests*. ERIC, 1993.
- [38] V. Raychev, P. Bielik, and M. Vechev. Probabilistic model for code with decision trees. In *Proceedings of the 2016 ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages, and Applications*, pages 731–747, 2016.
- [39] K. Rivers, E. Harpstead, and K. Koedinger. Learning curve analysis for programming: Which concepts do students struggle with? In *Proceedings of the 2016 ACM Conference on International Computing Education Research*, pages 143–151, 2016.
- [40] E. Rowe, J. Asbell-Clarke, R. S. Baker, M. Eagle, A. G. Hicks, T. M. Barnes, R. A. Brown, and T. Edwards. Assessing implicit science learning in digital games. *Computers in Human Behavior*, 76:617–630, 2017.
- [41] D. Selent, T. Patikorn, and N. Heffernan. Assistments dataset from multiple randomized controlled experiments. In *Proceedings of the Third (2016) ACM Conference on Learning@ Scale*, pages 181–184, 2016.
- [42] Y. Shi, Y. Mao, T. Barnes, M. Chi, and T. W. Price. More with less: Exploring how to use deep learning effectively through semi-supervised learning for automatic bug detection in student code. In *In Proceedings of the 14th International Conference on Educational Data Mining (EDM) 2021*, 2021.
- [43] Y. Shi, K. Shah, W. Wang, S. Marwan, P. Penmetsa, and T. Price. Toward semi-automatic misconception discovery using code embeddings. In *LAK21: 11th International Learning Analytics and Knowledge Conference*, pages 606–612, 2021.
- [44] D. Shin, Y. Shim, H. Yu, S. Lee, B. Kim, and Y. Choi. Saint+: Integrating temporal features for ednet correctness prediction. In *LAK21: 11th International Learning Analytics and Knowledge Conference*, pages 490–496, 2021.
- [45] Y. Su, Q. Liu, Q. Liu, Z. Huang, Y. Yin, E. Chen, C. Ding, S. Wei, and G. Hu. Exercise-enhanced sequential modeling for student performance prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.
- [46] V. Swamy, A. Guo, S. Lau, W. Wu, M. Wu, Z. Pardos, and D. Culler. Deep knowledge tracing for free-form student code progression. In *International Conference on Artificial Intelligence in Education*, pages 348–352. Springer, 2018.
- [47] M. L. Swartz and M. Yazdani. *Intelligent tutoring systems for foreign language learning: The bridge to international communication*, volume 80. Springer Science & Business Media, 2012.
- [48] N. Truong, P. Roe, and P. Bancroft. Static analysis of students’ java programs. In *Proceedings of the Sixth Australasian Conference on Computing Education-Volume 30*, pages 317–325. Citeseer, 2004.
- [49] K. VanLehn. Student modeling. *Foundations of Intelligent Tutoring Systems*, 55:78, 1988.
- [50] K. VanLehn. The Behavior of Tutoring Systems. *International Journal of Artificial Intelligence in Education*, 16(3):227–265, 2006.
- [51] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.
- [52] L. Wang, A. Sy, L. Liu, and C. Piech. Learning to represent student knowledge on programming exercises using deep learning. In *In: Proceedings of the 10th International Conference on Educational Data Mining (EDM 2017)*, 2017.
- [53] S. Wang, A. Ororbia, Z. Wu, K. Williams, C. Liang, B. Pursel, and C. L. Giles. Using prerequisites to extract concept maps from textbooks. In *Proceedings of the 25th ACM international conference on Information and Knowledge Management*, pages 317–326, 2016.
- [54] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*, pages 2048–2057. PMLR, 2015.
- [55] M. Yudelson, R. Hosseini, A. Vihavainen, and P. Brusilovsky. Investigating automated student modeling in a java mooc. In *In Proceedings of the 7th International Conference on Educational Data Mining (EDM) 2014*, 2014.
- [56] M. V. Yudelson, K. R. Koedinger, and G. J. Gordon. Individualized bayesian knowledge tracing models. In *International Conference on Artificial Intelligence in Education*, pages 171–180. Springer, 2013.
- [57] L. Zhang, X. Xiong, S. Zhao, A. Botelho, and N. T. Heffernan. Incorporating rich features into deep knowledge tracing. In *Proceedings of the Fourth (2017) ACM Conference on Learning @ Scale*, pages 169–172, 2017.
- [58] G. Zhou, H. Azizsoltani, M. S. Ausin, T. Barnes, and M. Chi. Hierarchical reinforcement learning for pedagogical policy induction. In *International conference on Artificial Intelligence in Education*, pages 544–556. Springer, 2019.

Building a Reinforcement Learning Environment from Limited Data to Optimize Teachable Robot Interventions

Tristan Maidment¹, Mingzhi Yu², Nikki Lobczowski³, Adriana Kovashka^{1,2},
Erin Walker^{1,2,3}, Diane Litman^{1,2,3},

Timothy Nokes-Malach³

¹Intelligent Systems Program, ²Computer Science Department, ³Learning Research and Development Center

University of Pittsburgh
tdm51@pitt.edu

ABSTRACT

Working collaboratively in groups can positively impact performance and student engagement. Intelligent social agents can provide a source of personalized support for students, and their benefits likely extend to collaborative settings, but it is difficult to determine how these agents should interact with students. Reinforcement learning (RL) offers an opportunity for adapting the interactions between the social agent and the students to better support collaboration and learning. However, using RL in education with social agents typically involves training using real students. In this work, we train an RL agent in a high-quality simulated environment to learn how to improve students' collaboration. Data was collected during a pilot study with dyads of students who worked together to tutor an intelligent teachable robot. We explore the process of building an environment from the data, training a policy, and the impact of the policy on different students, compared to various baselines.

Keywords

Reinforcement learning, teachable robots, balance of participation, lexical entrainment

1. INTRODUCTION

Pedagogical agents have demonstrated the potential to positively impact student learning and motivation in an educational setting [38, 39]. They are commonly characterized as either physical robots or virtual agents and provide an extra opportunity for students to have a social, interactive, and personalized learning experience. Pedagogical agents can be scaled by deploying multiple instances for students to use. While robots are not as easily deployed as virtual agents, recent research suggests that robots can be more effective than virtual agents in engaging students [32]. There is increasing interest in understanding how robots might play a role in formal and informal learning environments [31, 40].

T. Maidment, M. Yu, N. Lobczowski, A. Kovashka, E. Walker, D. Litman, and T. Nokes-Malach. Building a reinforcement learning environment from limited data to optimize teachable robot interventions. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 62–74, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853129>

One potential advantage of a robotic agent over a virtual agent is its ability to physically interact with the students. Some work has explored the use of robots capable of facial expressions and found its behavior increased the learning efficiency of students [51]. It had been observed that the physical embodiment of a robotic agent resulted in a higher level of social influence [46]. Other work has explored using gestures from a humanoid robot to help provide feedback to students about multiplication table problems [30].

Robots can take on multiple roles in interactions with a learner, including a tutor, a peer learner, or a tutee [4]. When robots take on the role of tutee, they are often called teachable robots. There are examples of success with teachable robots and a single student [23, 59, 37]. However, there has been less exploration of the use of teachable robots in a collaborative learning setting [12], with a student dyad [58]. This is a limitation of prior work since student collaboration is a powerful tool for improving learning [8, 14, 41]. Collaboration among peers in a computer-supported group setting can be supported by specially designed tasks that encourage interaction. [13]. Computer-supported group settings can encourage collaboration by targeting specific areas and difficulties of group interaction [57].

There are many ways that pedagogical agents adapt and personalize to students. One standard method is a statistical approach, such as multi-armed-bandit [11, 53], but these approaches have some trade-offs [47]. These systems require task-specific expert-authored rules; this may not be a sustainable approach for multi-disciplinary use. In contrast, a data-driven and automated approach is an exciting new way of understanding how to implement personalization.

Reinforcement learning (RL) is a data-driven approach for learning a policy for interacting with the students [44]. While multi-armed bandit approaches identify one action as the best and only select that action, RL policies use context about their environment to choose the best action for a specific moment¹. Traditionally, reinforcement learning uses a

¹We note that *contextual* multi-armed bandits exist. However, they make the assumption that the actions picked have no effect on the environment (in this case, the students). We aim to model how the students' behaviors change over time in response to the actions selected, and thus a contextual multi-armed bandit does not fit this problem.

virtual environment, which is used to train an initial policy, that is then transferred to the real world.

The RL-controlled robotic tutor described in Park et al. [44] does not involve the use of a virtual environment to train, instead training directly on the students. The experiences collected are always using the latest iteration of the policy, which is continually changing. While real-world RL ensures that the students interact with the most up-to-date policy, it actively uses students to test potentially sub-optimal actions during training, which might lead to confusion and unnatural interactions. With this approach, the policy learns through extensively interacting with students, requiring many months of training and needs to build a profile about each student. Real-world RL is also well-known as being particularly challenging [16].

Learning an RL policy in a virtual environment has two main benefits over real-world reinforcement learning. A policy can be trained in just a few hours by simulating millions of different versions of the environment and their outcomes. Furthermore, by training solely in a simulation, there is no need to train a policy through extensive experimentation on students. However, building such a simulation of the real world is challenging. One solution is offline reinforcement learning, in which an algorithm learns from a set of previously collected data, which obviates the need for a simulated virtual environment. Unfortunately, offline reinforcement learning requires an extensive collection of saved experience data [17], which is infeasible to collect from students.

We, therefore, assume a hybrid approach, using the set of previously collected data to model and build a data-driven simulation, effectively allowing for online reinforcement learning in an offline reinforcement learning setting. We explore the process of building a high-quality data-driven virtual environment, the difficulties in modeling the students, and ensuring environmental continuity.

We aim to see how a pedagogical robot can assist students in a group setting. The humanoid robot interacts with two students concurrently, as a social learning companion. We attempt to use Reinforcement Learning to aid in the decisions made by the teachable robot, to improve student learning outcomes and motivation by supporting students in a personalized manner.

The robot, named Emma, interacts with students via natural language. As the students work out the multi-step problems, they explain their solutions to Emma. Emma responds to the students' solutions, clarifying the purpose of the most recent step or occasionally asking thought-provoking follow-up questions. These actions are combined with gestures to make interacting with the robot feel more natural.

Student interactions with Emma were collected using an expert-authored natural language dialogue script. We built a simulation describing the interactions between the students and modeled the different types of interactions those students had with Emma. The simulation uses collected data to represent various types of student groups.

Extensive testing of the policies trained and tested in the

simulation demonstrates that RL methods outperform the dialogue script baseline in terms of collaboration metrics. Furthermore, in our simulation, we find that the RL methods improve collaboration among all student groups, unlike our baseline method, which fails to improve collaboration for some groups. Our contributions include:

- A hybrid approach to perform offline RL in a robot-student setting using online RL trained in a virtual environment.
- A novel method for building a virtual environment from previously collected data, to simulate student collaboration.
- An early exploration of applying RL for a teachable robotic agent in a collaborative setting with students.

2. BACKGROUND

Reinforcement Learning (RL) has provided recent breakthroughs in solving complex machine learning tasks, highlighted by applications in self-driving vehicles [29], super-human performance in competitive board and video games [54, 55], and advanced autonomous robotic control [1, 42]. RL allows an agent or policy to sequentially select actions to achieve some (potentially non-differentiable) goal.

Reinforcement Learning differs from the more traditional supervised learning, where a classifier is exposed to a set of labeled examples, with the goal of correctly classifying an unseen example. There is often no label or ground truth for what actions are right or wrong for an RL agent to select in the settings where reinforcement learning is used. Additionally, each action may have cascading effects on the environment's future, and similar actions may have varying effects that depend on the current state of the environment.

The policy in RL is a function, typically modeled with a neural network, that selects actions to navigate some environment given information about the current state of the environment and a reward that grades the actions. In the described reinforcement learning scenarios above, such as self-driving, competitive gaming, and autonomous robots, an agent is typically trained from scratch in a simulated environment. This policy, trained in the simulation, is then transferred to the real world.

For example, consider the case of autonomous robots. A realistic virtual environment, like those created with video game engines, is used, with an accurate model of the robot and the various motors involved in its control [7]. The policy can then control the motors of the simulated robot to try and attain some reward or goal. Transferring a policy from a simulated space to the real world is easier when the simulation is similar to the real world, emphasizing the quality of the simulated environment to decrease the domain gap [64].

Constructing an environment requires a deep understanding of how actions cause the environment to respond. In some settings, such as competitive chess playing, this is trivial: strict rules define how the pieces move and there is no stochasticity that affects the outcome of each action. In the case of self-driving, the actions and how the environment responds have a measure of stochasticity. While turning the

wheel to the left typically turns the vehicle to the left, features of the environment, e.g. a patch of ice on the road, can affect the car’s trajectory in a unintended manner.

In our setting, the effects of a policy’s actions on the simulated environment are not easily defined. For example, similar actions performed by the policy may vary in their outcome due to differences between students. Even when asking two different students the exact same question, their responses will most likely differ. The environment, therefore, must use the collected data to model many potential outcomes for actions taken.

2.1 Education and Reinforcement Learning

Reinforcement learning in education has been a topic of exploration since the early 1960s [24], specifically for the sequencing of instructional activities. Various approaches have been built on this concept over the years [3, 33, 11]. A survey performed by Doroudi et al. found that 21 of 36 studies of RL-sequenced instructional policies determined that the RL policy outperformed baseline policies [15].

Singla et al. present a recent survey of the use of RL in education [56], which identifies five main research directions of RL in education. Singla found RL was used for: personalizing curriculums [15, 48], personalizing hints/scaffolding/quizzing [2, 62, 45], A/B testing educational methods [34, 63, 49], modeling human students [22, 60, 26], and content generation [28, 27, 20]. Our work combines personalizing constructive feedback and modeling human students. In our setting, where we attempt to maximize collaboration between students, we model the students’ interactions regarding the feedback and hints provided.

2.2 Emma: The Teachable Robot

Our study context involves Emma, a humanoid NAO robot who interacts with a pair of students tasked to solve multiple ratio word problems, using natural language and basic gesture. NAO robots are commonly used and have been effective in social robot student learning settings [25]. The students work together to solve the problem and teach the solution to Emma. This collaboration has two benefits. First, we can obtain more data regarding student thinking, as they must vocalize more between themselves, compared to when working alone. Second, there are known benefits for student collaboration for improving student learning [8, 14].

During a study session, each question is shown to the pair of students, who need to solve the problem and then describe to Emma how to calculate the solution. The problems require multiple steps to solve, and the students need to convey each step to Emma. If one of the steps is wrong, Emma is designed to either ask a question to guide the students in the right direction or make a statement regarding the students’ response. An example interaction between students and Emma is shown below:

1. **Emma:** So I know that I need 66 hours worth of batteries and I know how much time I need for one battery.
2. **Student B:** So we can tell her 66 hours divided by 3 and three quarter hours.
3. **Student A:** Yeah, that’s right

4. **Student B:** So you’re going to divide 66 hours by 3 and three quarter hours.
5. **Emma:** Since we need to figure out what times 3 and three quarters is 66. So then 66 divided by 3 and three quarters is 17.6 Is that it?
6. **Student B:** Yes, that is correct.
7. **Emma:** I need 18 batteries because I need more than 17. I will put the answer as 18.

Students use an in-house web interface to see the math problem that the robot was solving, which provides controls to advance to the next step, a button to “push to speak” with the robot, and to see what the robot most recently said (as a backup to hearing it over the video chat service). The video feeds from each session (group) were recorded, along with the audio from each participant. The audio feeds for each participant were transcribed using IBM Watson’s Speech-to-Text system, with word timings maintained.

Our work follows two theoretical frameworks for identifying productive collaboration. First, the interactive-constructive-active-passive (ICAP) framework of cognitive engagement [10] hypothesizes that students collaborate best when actively interacting, taking turns to contribute constructively to the task at hand. In our environment, this would involve the students both working to teach Emma, participating relatively equally and engaging in dialogue with each other rather than pursuing independent parallel work. Second, we draw on the Interactive Alignment Model (IAM), which postulates that in order for two collaborators to align their understanding, they must align their communication [19]. One indication of this alignment is lexical entrainment, which is indeed related to student success in group learning settings [18]. Lexical entrainment measures the similarity between the language used by speakers in a group over time [6], and might be a sign of students converging to a shared mental model [18], which can lead to successful group performance [61]. We are interested in exploring both balance of participation and lexical entrainment as rewards in an RL environment.

2.3 Data Corpus

Due to the COVID-19 pandemic, sessions were held using an online video-chat service to connect the students with Emma. For our pilot studies, we collected data from twenty-eight undergraduate students who interacted with the robot in groups of two (i.e., 14 dyadic sessions; 11% Male, 89% Female; 32% Asian, 14% Black, 46% White, 7% no response; Mean age = 19.4 years, SD = 1.19)². Each session lasted approximately 30 minutes.

In addition, students were individually tested and surveyed before and after collaboratively interacting with the robot. These assessments were used to determine pre-existing (i.e., pre-collaboration) factors that may impact mid-collaboration interactions and participation and post collaboration learning outcomes. The pre-collaboration measures included: motivation (i.e., interest, utility, efficacy, attainment, and cost), goal orientations (i.e., mastery approach, performance approach, performance-avoidance), attitudes towards robots

²This data is part of a more extensive study that includes another condition in which individuals taught Emma. For the purposes of the current work, we focus on the dyads.

and collaboration (i.e., work quality, peer support, interdependence), affinity for technology, and prior knowledge (i.e., pretest on ratios). The post-collaboration measures included ratings of rapport with their partner and Emma (i.e., general, positivity, or attentiveness), perceptions of Emma (i.e., anthropomorphism, likeability, animacy, and intelligence) [36, 35], and posttest scores (i.e., counterbalanced with pretest). Given the need to consider the group as the unit of analysis, we used the dyads’ average measures for our analyses, with higher averages representing a larger presence of each construct. In total, 27 metrics were collected from the surveys and assessments. This study was approved by the Institutional Review Board.

3. METHODOLOGY

We construct our simulated environment using real-world data collected from interactions with real students. The environment models the conversational interactions between all participants and uses probabilistic methods to capture the various sources of stochasticity. This allows us to model *potential* outcomes that could occur when Emma interacts with the students, even when such an interaction was not captured in the pilot study. We then train a set of three RL algorithms using the environment and compare the results of each algorithm with a series of baselines.

3.1 Data and Setup

Building an end-to-end RL environment is highly task-specific but can be generally simplified into two steps. We first extract the data gathered during the student interaction sessions. Next, we use the extracted data to model student interaction and observations in the environment in conjunction with various probabilistic sequence modeling techniques. This process involves identifying reward metrics and modeling how the environment responds to stimuli.

3.1.1 Student Participation Metrics

Our data-driven simulation uses various metrics captured during the data collection studies. Specifically, we capture the change in student participation metrics after Emma performs an action. We observe how these metrics change throughout the study as the students interact with each other and the robot. We use metrics that approximate the balance of interactive participation (following the ICAP framework) and lexical entrainment between the students. For our environment, we use Measure of Participation and Word Co-Occurrence metrics, respectively.

The text transcripts were tokenized at the sentence level using NLTK [5]. The Measure of Participation and Word Co-Occurrence were calculated as shown below, and associated with each sentence. This methodology allows us to estimate the balance of participation and lexical entrainment at intermediate points of the session.

Measure of Participation (MoP) MoP indicates the balance of group participation as proposed by Paletz and Schunn [43]. MoP computes the average level of involvement in the group, scaled between 0 (equal) and 1 (dominated) participation. MoP is defined in Equation 1, where \bar{n} is the average number of people during the session, N is the max number of people present (always 2 in our setting), M is the total

number of utterances said in the session, n_k is the number of people present on utterance K (always 2), i is the index representing a student, and m_i is the number of utterances where person i is present.

$$P_s = \bar{n}^2 * \frac{\sum_{i=1}^N |\sum_{K=1}^M f(n_k, i, K)|}{2(\bar{n} - 1) \sum_{i=1}^N m_i} \quad (1)$$

$$f(n_k, i, K) = \begin{cases} \frac{n_k - 1}{n_k} = \frac{1}{2} & \text{if } i \text{ is speaking during utt. } K \\ \frac{-1}{n_k} = -\frac{1}{2} & \text{if } i \text{ is silent during utt. } K \end{cases}$$

For example, given two utterances and one participant speaks during each, $MOP=4*(|1/2-1/2|+|-1/2+1/2|)/[2*(2+2)]=0$. If only the first participant speaks during each utterance, $MOP=4*(|1/2+1/2|+|-1/2-1/2|)/[2*(2+2)]=4*2/8=1$.

This metric is invariant to changes in group size and supports groups of different sizes. In our RL environment, since the agent’s goal is to maximize the reward, we use $1 - P_s$ for the feedback. This, in turn, rewards the agent for achieving equal participation.

Word Co-Occurrence (WCO) WCO provides a simple estimate of lexical entrainment. WCO is defined as the number of lemmas common between both participants. Lemmatization was performed using WordNet in NLTK. To get WCO in the same range as MoP, the value is normalized between 0 and 1 by dividing by the number of unique words said in the session. The size of the shared set of words increases only if both participants are contributing and talking about similar material.

3.2 Overview of the RL Environment

A reinforcement learning environment requires four essential components. The first is the set of potential actions A that the agent can take at each step. The second is the set of states that the environment can take S . The last is the set of rewards R that the agent will be attempting to maximize.

The first step to building an environment is defining each of the required components of the environment and at what level of abstraction these components will be represented. We define how the environment responds when the robot interacts with it, the information provided by the environment to the robot, and the options the robot can take at each step.

3.2.1 States

The environment state is used to describe to the RL policy what is happening at a specific point in time. In this case, we model the environment state as a representation of the sentence said directly to Emma by one of the students - this is what Emma observes and uses to select her next action. For example, a student may tell Emma, “If you use 3/4 of your battery in 1 hour, you multiply that by 3 to figure out how much you use in 3 hours”.

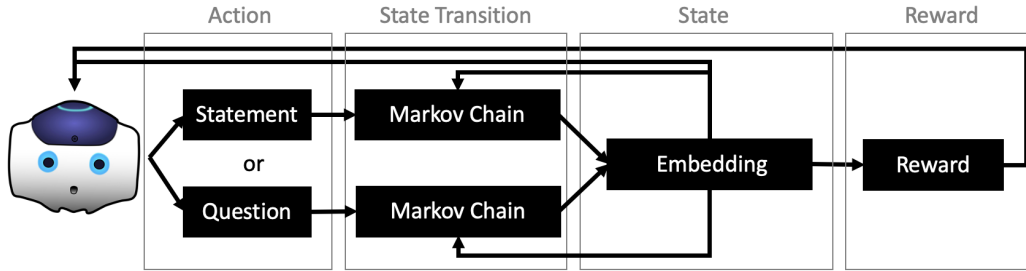


Figure 1: A high-level overview of the different parts of the environment. Emma’s actions each use a Markov chain, trained using collected data, to transition between states. The current state of the environment conditions the state transition. States are represented as sentence embeddings.

3.2.2 Actions

In the case of determining the action space of the environment, we look at what Emma was capable of during the data collection sessions. As a proof of concept for our reinforcement learning approach, we simplify the action space to include the two general responses – questions and statements. For example, one response from Emma is this question-based reply: “So if I have forty-five hot dogs and I now know how much it costs for one, can I figure out how much it costs for forty-five hot dogs by multiplying?”. For the statement-based reply for the same problem, she says, “With the unit rate, I know it costs two dollars and fifteen cents for each hot dog. I don’t remember what I am supposed to do with that, though.” As we can see, while the responses convey similar information about the problem, they could potentially invoke very different responses in the student participants.

3.2.3 State Transitions

In addition to defining the states and actions of the environment, we need to define how the actions cause the states to transition from one to another. Our state transitions describe how the students react to Emma’s responses. Between Emma’s actions, there are a series of interactions between the students, where they decide what to say to Emma or work on the problem solution. We model the progression of these inter-student interactions probabilistically to describe how the students’ conversations lead to different interactions with Emma.

3.2.4 Reward

The purpose of the reward is to coerce the policy towards performing the actions needed to achieve the required outcome. There are broadly two ways to reward the agent: at constant intervals throughout a session or once at the end, but the latter sparse option is more complex than frequent rewards [21]. Our student participation metrics (Sec. 3.1.1), derived from learning theory hypotheses, can be calculated at intermediate points during the session. To keep the environment simple, we give an intermediate reward after each of the robot’s actions. The value of the reward is the sum of the student participation metrics associated with the current state (MoP + WCO), as observed during data collection.

3.3 Simulation

Now that we have presented a high-level view of the environment that was designed, we will delve into the technical

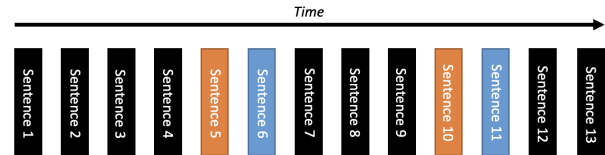


Figure 2: An example of the interactions between the students and Emma. The black blocks signify a sentence said by a student to another student, the orange blocks a sentence said to Emma, and blue blocks a response to the students.

details about how it is implemented. In the sections below, we highlight three difficulties in implementing a data-driven environment from previously collected data. First, the conversations between the students can vary significantly but need to be accurately modeled. Second, with data collection limits, we must handle missing information when simulating the outcome of actions. Third, the simulation must provide a reward to provide feedback for the selected actions.

3.3.1 Simulation Design

An overview of the environment is shown in Figure 1. Emma is capable of two responses, to either reply with a question or a statement. Each action has a unique Markov chain, which generates a sequence of simulated interactions between students. The generation of the simulated interactions is dependent on the current state of the environment and is used to generate the subsequent state. This subsequent state is associated with a reward for Emma.

In Figure 2, we depict an example of the structure of the collected data, with time represented on the x-axis. Here, the black blocks signify a sentence said by a student to another student, the orange blocks represent a sentence said to Emma, and the blue blocks indicate Emma’s response to the students. The students say four sentences among themselves before one student says a sentence that is an input to Emma. The 6th block is the reply Emma chooses for the students. The students talk between themselves for three sentences before interacting with Emma again. Emma responds, the students continue talking, and the cycle repeats.

We can see that the interactions between the students have a structured turn-taking nature, which follows a repeated

pattern: 1. students converse between themselves, 2. students provide an input to Emma, 3. Emma replies to the student’s input. Using this general structure of interaction, we can define the necessary portions of the environment. The state of the environment, as observed by the RL policy, is the sentence that is said directly to Emma. The actions in the environment are the replies from Emma to the students. The rewards of the environment are the changes in the levels of the Measure of Participation and Word Co-Occurrence scores that occur as the students discuss. The rewards are provided at the end of each sentence. The state transitions occur due to Emma’s responses (actions) to the students - this invokes conversation between the students, resulting in a specific response from the student (state).

3.3.2 State Space (S)

The first step in building the environment is setting a level of abstraction to represent the states, S . This step is important for modeling the environment input to the RL agent and is necessary in describing how the environment transitions from one state to the next. The interactions between the students and Emma can be modeled using the input sentence tokens, which are the orange blocks in Figure 2.

The tokens are encoded using Google’s pre-trained Universal Sentence Encoder [9]. The Universal Sentence Encoder provides embedding vectors of each token, for the tokens said between the students and the tokens said as input to Emma. Embedding vectors are high-dimensional representations in a semantic space. The embeddings of semantically similar tokens will have a higher cosine similarity score than semantically different tokens.

Let $p \in P$ be a sentence in the set of sentence tokens said as input to Emma, collected during data collection, and $encode(x)$ converts a token to a 512-dimensional embedding with the Universal Sentence Encoder. A state in the environment is defined as $s \in S$, where $s = encode(p), \forall p \in P$.

3.3.3 Action Space (A)

In Figure 2, Emma’s “actions” are her natural language replies shown as the blue blocks. For simplicity, Emma has the same set of actions that can be taken at each step. The replies to the students are roughly split into two categories: question-based replies and statement-based replies. An action $a_{t,p} \in A$ at timestep t has an indicator, p , to describe the type of action taken. p can be one of (q, s) , i.e. Emma can reply with a question $a_{t,q}$ or a statement $a_{t,s}$.

3.3.4 State Transitions

One vital mechanism for an environment is the *state-action transition*. The state-action transition defines for some state s_t and action a_t at time step t , the next state s_{t+1} emitted by the environment, at time step $t + 1$. Referring again to Figure 2 (black-colored sentence tokens), we see that an action is followed by dialogue between students, which takes place when transitioning between states.

We model this state-action transition using the Markov chain, a probabilistic sequence model. We choose to use the Markov chain for this simulation due to its simplicity and ability to model time-series data. A Markov chain is a simple model

where the probability of the n^{th} event $e_{t,n}$ at time t , is conditioned and only dependent on the previous event $e_{t,n-1}$. Here, each event is a sentence uttered by a student, and the chain represents the transitions of the conversation had between the students before they converse with Emma.

To build the Markov chain, we must simplify the events from the continuous space (sentence embeddings) to a countable space. With semantically similar sentences existing close together in the embedding space, unsupervised clustering methods allow for an intelligent way to represent the sentence tokens in a countable space. The embeddings for the transitional sentences are assigned to clusters in an unsupervised manner via KMeans clustering. Each sentence is assigned and represented by a cluster c via this method. The set of clusters $C, c \in C$ is the countable space for the Markov chain. We denote here a function $c_{t,n} = k(e_{t,n})$ which for event embedding $e_{t,n}$, returns the cluster identifier of each embedding, $c_{t,n}$. For simplicity, we extend $k(x)$ to support operations on a sequence of events, which it converts to a sequence of cluster identifiers in the same order.

The state-action transition sequences collected from the pilot study are used to build the chains. For each state-action transition sequence, we store the action a_t , the starting state s_t , the transitional sequence of sentence embeddings $E_t = e_{t,1}, e_{t,2}, \dots, e_{t,n}$ and ensuing state s_{t+1} . The transitional sentences, and ensuing state are combined as a sequence of embeddings $\hat{E}_t = e_{t,1}, \dots, e_{t,n}, s_{t+1}$. This sequence of embeddings is then transformed to be represented in terms of each sentence’s cluster identifier, $H_t = k(\hat{E}_t)$. We also define another function, $j(c)$, which randomly samples and returns a sentence (its embedding) from cluster c . These sequences of cluster identifiers are used to build a Markov chain. A separate Markov chain is built for each action to separate the state transitions of the two types of actions. These action-specific Markov chains are m_q and m_s , respectively.

State transitions are modeled using a conditional generation of the Markov chains. Sequences generated by Markov chains are typically initialized randomly by selecting a random event as the starting point. Conditional generation involves selecting a specific event as a starting point, which affects the subsequent generation of events in a cascading manner. This conditioning ensures the actions selected by Emma in the virtual environment affect the following simulated interactions between the students. To describe conditional generation, we provide the example $m_q(e_{t,1})$, to represent the conditional generation of the next state using the *question* Markov chain. The output of $m_q(e_{t,1})$, is a sequence of cluster identifiers H_t , that starts with cluster identifier $k(e_{t,1})$.

Mappings, $r_{s,a} = [(s_t, a_t) \rightarrow e_{t,1}]$, are extracted from the recorded state-action transitions, to be used for the data-driven conditional generation. For an observed state s_t with action a_t , the Markov chain conditionally generates with starting point $k(e_{t,1})$, and ends with some cluster identifier $c_{t,n}$. For example, we generate the following sequence of events, $E_t = e_{t,1}, e_{t,2}, \dots, e_{t,n}$, and use the last event to obtain our sampled cluster identifier, $c_{t,n} = k(e_{t,n})$. This cluster is used to sample next state, $s_{t+1} = j(c_{t,n})$. We denote the set

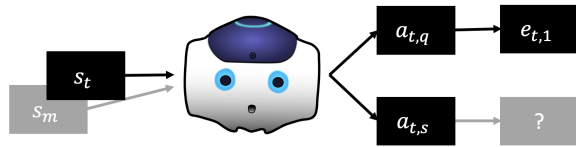


Figure 3: A scenario where an action’s outcome is not contained in the collected data. To approximate the effect of the action, we substitute in a similar state-action transition from the collected data in a probabilistic manner.

of recorded state-action transition mappings R .

State-Action Map Substitution The data-driven mappings, $r_{s,a} \in R$, are a straightforward way to conditionally generate the next state, for some observed set of state and action, (s_t, a_t) . However, consider the example in Figure 3. During data collection, we observed and recorded the mapping $(s_t, a_{t,q}) \rightarrow e_{t,1}$, shown in the top row. However, we may, during the simulation, want to test what happens for the state-action set $(s_t, a_{t,s})$, for which no recorded mapping exists. With no existing mapping, we need to find a substitute - a recorded mapping with a state that is similar to s_t , that instead contains action $a_{t,s}$.

We propose to find a substitute mapping by filtering the set of all recorded state-action transition mappings R . Within R , we filter and keep recorded mappings that contain the desired action, $a_{t,s}$ in this example. For each mapping in the filtered set, we calculate the cosine similarity score $sim_{t,m} = \cos(s_t, s_m)$ between the current state s_t of the environment, and the state s_m stored in the mapping. This similarity score is used to assign a probability for being selected as the substitute mapping to each mapping in the filtered set. The probabilities are proportional to the similarity score so that the most similar states have the highest probability of being selected. The filtered set is then sampled using the probabilities to find a new mapping, with a similar state s_m , the selected action $a_{t,s}$, and a new substitute event for conditional generation $e'_{t,1}$.

3.3.5 Reward (R)

The formulation of the environment reward is simple since each entry in the Markov chain has an associated change in WCO and MoP that was observed during data collection. Therefore, during a state transition, we aggregate the WCO and MoP across the sequence generated by the Markov chain while generating the next step. The sum of the rewards over the chain is the reward provided for the action.

As a reminder, each entry in the chain is a specific cluster identifier in which various sentences reside. We use the cluster’s mean reward (using all the sentences in the cluster). As an example, if Emma selects a question as her action, we sample the Markov chain, $m_q(e_{t,1})$, to generate the next state s_{t+1} . This generates the chain: $E_t = e_{t,1}, \dots, e_{t,n}$ where $s_{t+1} = j(k(e_{t,n}))$. Each entry, $e_{t,m}$, in the chain has an associated reward (WCO + MoP). Let $r_{t,m}$ be the average reward for cluster $c_{t,m}$, where $c_{t,m} = k(e_{t,m})$. To get an action’s full reward, we compute the sum of the associated rewards along the sampled chain $R_t = \sum_{m=1}^n r_{t,m}$.

Time Constraints To ensure that the environment does not provide rewards that bias policies towards selecting actions that have longer Markov chains, we add a time limit to how long an environment can run. The sentences said during data collection are timestamped, which describes how much time was required to say each sentence. Throughout the simulation, a timer is kept, and if a sentence is sampled by the Markov chain, the timer increments by the amount of time required to say the sentence, using the word timings extracted from the automated Speech-to-Text system. The time limit is a 30-minute cutoff, which mirrors the cap to session length that was in place during data collection. Any reward obtained after the cutoff is excluded.

3.3.6 Additional Implementation Details

Random Student Initialization The environment is designed to create a policy that is capable of handling different types of learners, different styles of student participation, and different levels of student interaction with Emma. To emulate this, during the environment initialization, Emma is blindly assigned to a random student group (one of the 14 from the pilot study); the policy is not provided information about which group was selected. Emma interacts with a version of the environment that uses state, state transition, and reward parameters collected from only the selected student group during that session. For subsequent sessions in the simulation, the student group is blindly and randomly re-assigned, ensuring that the learned policy can generalize to multiple types of students. These parameters differ per group, meaning the environments respond quite differently, depending on the current student group being represented.

Environment Tuning The first step, before training the Reinforcement Learning policy, is to make sure that the environment works as intended given the assumptions we used to model the states, actions, rewards, and state transitions. To do so, we built a deterministic policy for each group, which selects actions exactly as were selected by Emma in the pilot study - we call this deterministic policy the *mirror policy*. To be specific, in the simulation, she *only* picks actions that have an **observed** state-action transition, so no action substitution is used. While there is some stochasticity in the environment that comes from sampling the Markov chains, the environment should respond, on average, similarly to the outcomes that were observed during the pilot study.

To calculate the environment similarity, we directly compare the mirror policy’s action distribution and the action distribution of the studies during data collection. This comparison is made by comparing the ratio of mirror policy’s action distribution and the study action distribution, using mean squared error (MSE). We call this metric the *action distribution difference*. A low action distribution difference indicates that the environment reacts as expected.

Policy Tuning We tested three standard methods for reinforcement learning, Deep Q-Network (DQN), Proximal Policy Optimization (PPO), and Advantage-Actor Critic (A2C). DQN is a Q-learning method where a neural network learns the value of actions for a particular state. PPO is a policy-gradient method, which estimates the policy gradients directly with respect to the reward. A2C is an actor-critic method, where the critic learns the value of a state, and

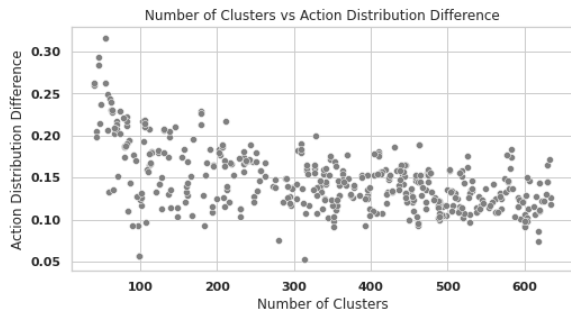


Figure 4: The *environment’s action distribution difference* varied with the k -value for the sentence KMeans clustering. After about $k = 400$, we see little change.

an actor to learn how to select actions. These policies are implemented using the Stable Baselines package [50].

4. RESULTS

Comparing RL Policy to Observed Decisions Once we identify the best RL policy, we compare the actions and rewards it achieves to those of the deterministic policy used during data collection, as a sanity check for our environment.

Comparing RL Policy to Random Baseline To see the performance of the learned policy against an unbiased policy, we compare the former against a random policy, which selects actions with equal probability. Consistently outperforming a random policy demonstrates that the learned policy learned to navigate the environment and correctly select actions.

Group Equality The amplitude of a policy’s reward in the environment is not the only goal for a successful policy. On top of maximizing the reward, we look to ensure *group equality*, i.e. positive reward across *all* groups - to verify that the learned policy helps different types of students.

Correlations of RL Policy Gains with Student Survey and Test Metrics The survey and assessment measures were not used in the environment design or parameters but could impact, or be impacted by, our RL metrics. Thus, we used the Pearson correlation coefficient to determine which dyadic factors (i.e., from the average survey and assessments metrics) were most highly associated with our RL Policy’s improvement in rewards. Doing so provides insight into how pre-existing factors connect to rewards during interactions with the robot and how these interactions connect to post-collaboration perceptions and assessments.

4.1 Evaluating the Environment

The mirror policy is used to tune the environment, and ensure that the environment works as expected. The number of clusters used for the sentence representation affects the level of stochasticity in the environment. This stochasticity is present in the language generated via the Markov Chains. An insufficient amount of clusters can result in a loss of the structure in the environment. With less clusters, the language generation becomes more random, and fails to capture the information from the pilot study. An excessive amount of clusters can cause the environment to respond *only* as was

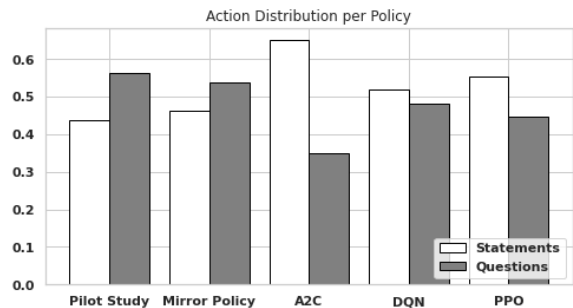


Figure 5: The action distributions of the various policies. In the simulated environment, the mirror policy acts similarly to the actions observed in the pilot study. The RL policies all tend to prefer statements.

observed during the pilot study, generating nearly-identical language to what was observed.

To determine the correct number of clusters, the mirror policy’s action distribution was evaluated while varying the number of clusters compared to the real distribution, as shown in Figure 4. Under 100 clusters, we see a large difference in how the environment responds to the mirror policy’s actions compared to the observed distribution. However, as we increase k , we see a decrease in the difference, with little improvement when $k \geq 400$. For that reason, we set $k = 400$ for all further experiments to ensure that there is some variability in the environment without deviating from what was observed in data collection.

In Figure 5, we display the action distribution of the mirror policy in the simulated environment compared to the observed real-world data. We can see the mirror policy acts quite similarly to what was observed in the real world – differences in actions come from the various sources of stochasticity in the environment.

4.2 Evaluating the Policy

Questions are used more often than statements in both the pilot study and the mirror policy, but *all* RL policies flip the action distribution around - preferring statements over questions, indicating that there may be deficiencies in the deterministic policy used for the pilot study.

Tuning and validating the policy were done via two types of cross-validation. We tested the environment using traditional 5-fold cross-validation, where the 14 groups are separated into five subsets or folds, and each fold is used as a testing set. This methodology models the scenario where Emma is trained then evaluated with multiple unseen groups of students. This setting aims to demonstrate that learned policies provide benefit to *more than one group*.

To further validate policies learned in the environment, we also test leave-one-out cross-validation, where each student group is used as a testing set. In this scenario, we model the event where Emma is trained in a simulated environment, then evaluated with a single unseen group of students. This setting is to demonstrate that learned policies provide benefit to *all groups*.

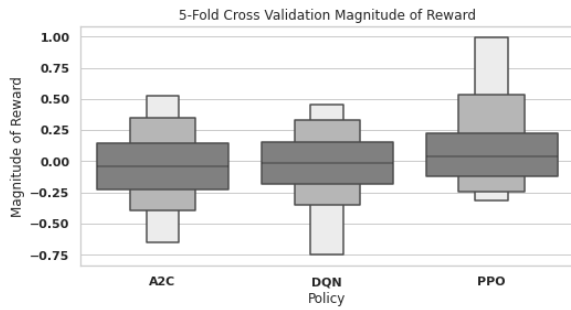


Figure 6: The difference in the performance of the RL policies and the mirror policy, per fold in K-fold cross-validation.

We compared various learning rates in both scenarios and three RL policy optimization algorithms, PPO, DQN, and A2C. In total, 7087 different policies were trained and tested. Cross-validation takes 75W of power and runs for approximately 35 minutes. Training was performed on 2 Nvidia GTX 1080ti (\$699 MSRP each). Data collection was performed with one NAO Robot (\$12990 MSRP).

4.2.1 RL Method Performance Differences

We refer to the difference in reward between an RL method and the mirror policy baseline as the *Magnitude of Reward*. In 5-fold cross-validation (Figure 6), the mirror policy outperformed the median A2C and DQN policies, as demonstrated by the slightly negative magnitude of reward. A2C’s magnitude of reward distribution is roughly normal, while DQN’s is negatively skewed. The median PPO policy outperformed the mirror policy, with a positive magnitude of reward. Furthermore, the PPO magnitude of reward distribution is positively skewed. PPO has an average magnitude of reward of 0.20, while A2C and DQN both have an average magnitude of reward of -0.11 and -0.10 , respectively.

In leave-one-out cross-validation (Figure 7), we see that the median magnitude of rewards for all RL policies is slightly negative. However, while A2C is negatively skewed, DQN is positively skewed - the average DQN magnitude of reward is slightly positive. The mean rewards of A2C, DQN, and PPO are -0.03 , 0.06 , and 0.02 , respectively. In both settings a higher learning rate resulted in an increase in the policy’s reward on average.

A2C under-performed the mirror policy in both cross-validation settings, indicating that it does not work well in this environment. While DQN outperformed PPO in leave-one-out cross-validation, it under-performed the mirror policy in 5-fold cross-validation, indicating a lack of stability between settings. PPO outperformed the mirror policy in both settings, demonstrating positive and stable performance.

4.2.2 Gains for Individual Groups

The magnitude of reward of the policy is not the only goal of using an RL policy. We also aim to help improve groups that performed poorly during the diagnostic testing so that there is equal performance and improvement among all groups. Gains among low-performing groups mean that the policy may potentially aid students who learn in non-standard ways.

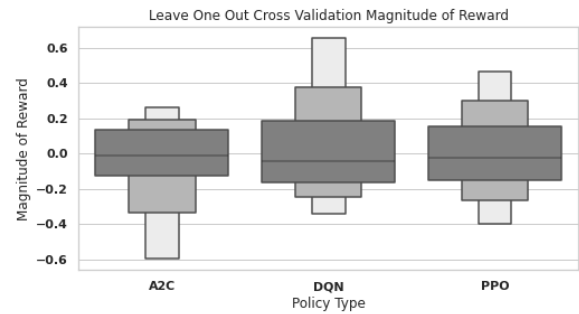


Figure 7: The difference in the performance of the RL policies and the mirror policy in leave-one-out cross-validation.



Figure 8: Average reward of the PPO RL policy in K-fold cross-validation, compared to the mirror policy and the random policy. The learned policies have a more consistent and positive average final reward.

Figure 8 shows that the learned PPO RL policies achieve a consistent and positive average final reward in cross-validation among all folds. Here, the mirror policy achieves very inconsistent results among folds and a negative reward for Fold 3. The random-action baseline outperforms the mirror policy here but achieves approximately 0 reward for Fold 3.

We see similar trends in leave-one-out cross-validation (Figure 9) but slightly more variation. Here, the PPO RL policies again achieve a positive reward, while the mirror policy and random-action baseline do not. The mirror policy and random baseline both achieved a negative reward for Fold 7, and the random baseline achieved a slightly negative reward for Fold 2. We note that Fold 2 generally did not respond significantly to any method, but the PPO RL policies achieved the highest average reward.

4.3 Correlations with Diagnostic Metrics

To determine the connections of our simulated environment to different dyadic factors, we calculated correlations between the Magnitude of Reward and metrics obtained by the questionnaire and assessments administered to students. These correlations may reveal which students are most positively affected by the boost in MoP and WCO that the learned policies can achieve in the simulated environment. We select the three diagnostic metrics with the highest Pearson correlation coefficients (in amplitude).

First, the large positive Pearson correlation ($+0.672$, $P =$

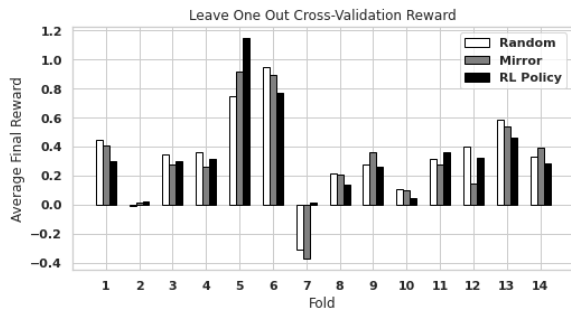


Figure 9: Average reward of the PPO RL policy in the leave-one-out cross-validation setting compared the random policy and mirror policy. The learned policies have a consistently positive average final reward.

.008) between the dyads’ average self-reported values for quality of work in groups and the Magnitude of Reward suggests that students who more highly value the quality of work in collaborative settings may have benefited more from the actions of the learned policies. We would expect to see this correlation, given the overlapping nature of the group-work quality construct and key components of Magnitude of Reward (i.e., balance of participation and entrainment).

Furthermore, the high negative correlation between the dyads’ average pretest scores ($-0.634, P = .015$) and the Magnitude of Reward of the policy may suggest that the groups with lower prior knowledge before the activity would likely have benefited more (i.e., better entrainment and balance of participation) from a policy that was trained in the reinforcement learning environment. Similarly, the high correlation between Magnitude of Reward and the dyadic-average understanding of ratios at posttest ($-0.781, P = .001$) implies that the groups who performed worse, on average, than their counterparts could have been benefited by the improved balance of participation and word co-occurrence from the reinforcement learning policies.

5. DISCUSSION AND CONCLUSION

Our tests show that our environment exhibits a similar behavior under the mirror policy as was observed during the pilot study, in terms of action distribution (Figure 5). Notably, the policies learned in the environment differ significantly from the pilot study and the mirror policy, heavily favoring statements over questions, opposite to the rule-based approach used for the mirror policy. One explanation is that a question occasionally resulted in a direct response from the student who originally spoke to Emma. We observed that statements more often resulted in the students reconvening and then discussing how to reply to Emma.

This assignment of actions may have its merits, based on its improvements in both the amplitude and consistency of environment reward. In both forms of cross-validation, the policies trained with PPO outperform the mirror policy. We note that other RL methods had inconsistent performance across different types of cross-validation. In the Proximal Policy Optimization algorithm paper [52], the author finds a improvement in performance *and sample efficiency* using

PPO over DQN and A2C. In our setting, we are limited in our number of data-driven samples, and therefore PPO’s strong performance may be from this data efficiency.

When looking at the results of the individual folds for both forms of cross-validation, we see that the PPO RL policies consistently achieve a net positive reward. In 5-fold cross-validation, RL’s rewards are consistent across folds, while the mirror policy and random baselines were inconsistent and even resulted in a significant decrease in one fold’s average reward. This trend extends to leave-one-out cross-validation, where the PPO RL policies again achieved a positive reward on all folds. From our testing, we believe that the PPO RL policies may benefit all students and demonstrate performance equity among groups.

These findings are further highlighted by the connections between the learned policies’ Magnitude of Reward and the metrics obtained by the questionnaire and students’ diagnostic tests. The Pearson correlations may reveal which students are most positively affected by the additional MoP and WCO that the learned policies can achieve in the simulated environment. The actions of the learned policies may help the most for students who care highly about their quality of work in a group setting and may provide the most benefit for underperforming students. Further empirical investigation into these links might reveal new understanding of how interventions under the ICAP and IAM frameworks interact with individual differences to improve collaboration.

However, the policies trained in this environment have not yet been validated in a real-world setting with students. Additionally, the pilot study was performed using a hand-crafted and deterministic script for interacting with the students. The data collected by the script’s actions may have biases, which could affect the accuracy of the environment. Furthermore, the pilot study data collection was performed with undergraduate school students, held virtually. We aim to test this system in-person and with middle-school students. We expect some of our findings in this study may not carry over to this new setting. In addition, our investigation only looked at one modality. In the future, we aim to extend this work, adding additional actions such as verbal references towards particular students and adding physical gestures and gaze to the robot.

The framework presented here provides a hybrid approach for using RL in a robot-student setting, by building an environment from collected student interaction data. This environment can be adapted to any number of actions and students without requiring extensive training in the real world.

Acknowledgements: This work was supported by Grant No. 2024645 from the National Science Foundation, Grant No. 220020483 from the James S. McDonnell Foundation, and a University of Pittsburgh Learning Research and Development Center internal award. We would also like to thank Mike Diamond for his help with the pilot study.

6. REFERENCES

- [1] F. Agostinelli, S. McAleer, A. Shmakov, and P. Baldi. Solving the Rubik’s cube with deep reinforcement learning and search. *Nature Machine Intelligence*,

- 1(8):356–363, Aug. 2019. Number: 8 Publisher: Nature Publishing Group.
- [2] V. Aleven, E. A. McLaughlin, A. Glenn, and K. Koedinger. Instruction Based on Adaptive Learning Technologies. 2016.
- [3] R. Atkinson. Optimizing the Learning of a Second-Language Vocabulary. *Journal of Experimental Psychology*, Vol.96(No 1):124–129, 1972.
- [4] T. Belpaeme, J. Kennedy, A. Ramachandran, B. Scassellati, and F. Tanaka. Social robots for education: A review. *Science Robotics*, 2018.
- [5] S. Bird, E. Klein, and E. Loper. Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit, 2009. original-date: 2009-09-07T10:53:58Z.
- [6] S. Brennan. Lexical Entrainment In Spontaneous Dialog, 1996.
- [7] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. OpenAI Gym. June 2016.
- [8] A. L. Brown and A. S. Palincsar. Guided, cooperative learning and individual knowledge acquisition. In *Knowing, learning, and instruction: Essays in honor of Robert Glaser*, pages 393–451. Lawrence Erlbaum Associates, Inc, Hillsdale, NJ, US, 1989.
- [9] D. Cer, Y. Yang, S.-y. Kong, N. Hua, N. Limtiaco, R. St. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar, B. Strope, and R. Kurzweil. Universal Sentence Encoder for English. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 169–174, Brussels, Belgium, Nov. 2018. Association for Computational Linguistics.
- [10] M. T. H. Chi and R. Wylie. The ICAP Framework: Linking Cognitive Engagement to Active Learning Outcomes. *Educational Psychologist*, 49(4):219–243, Oct. 2014.
- [11] B. Clement, D. Roy, P.-Y. Oudeyer, and M. Lopes. Multi-Armed Bandits for Intelligent Tutoring Systems. *Journal of Educational Data Mining*, 7(2):20–48, June 2015. Number: 2.
- [12] P. Dillenbourg. Chapter 1 (Introduction) What do you mean by ‘collaborative learning’? *Collaborative-learning: Cognitive and Computational Approaches*, Vol. 1, Jan. 1999.
- [13] P. Dillenbourg, S. Järvelä, and F. Fischer. The evolution of research on computer-supported collaborative learning. In *Technology-enhanced learning*, pages 3–19. Springer, 2009.
- [14] W. Doise, G. Mugny, and A.-N. Perret-Clermont. Social interaction and the development of cognitive operations. *European Journal of Social Psychology*, 5(3):367–383, 1975. Place: US Publisher: John Wiley & Sons.
- [15] S. Doroudi, V. Aleven, and E. Brunskill. Where’s the Reward?: A Review of Reinforcement Learning for Instructional Sequencing. *International Journal of Artificial Intelligence in Education*, 29(4):568–620, Dec. 2019.
- [16] G. Dulac-Arnold, N. Levine, D. J. Mankowitz, J. Li, C. Paduraru, S. Gowal, and T. Hester. Challenges of real-world reinforcement learning: definitions, benchmarks and analysis. *Machine Learning*, 110(9):2419–2468, Sept. 2021.
- [17] D. J. Foster, A. Krishnamurthy, D. Simchi-Levi, and Y. Xu. Offline Reinforcement Learning: Fundamental Barriers for Value Function Approximation. Nov. 2021.
- [18] H. Friedberg, D. Litman, and S. B. F. Paletz. Lexical entrainment and success in student engineering groups. In *2012 IEEE Spoken Language Technology Workshop (SLT)*, pages 404–409, Miami, FL, USA, Dec. 2012. IEEE.
- [19] S. Garrod and M. J. Pickering. Joint Action, Interactive Alignment, and Dialog. *Topics in Cognitive Science*, 1(2):292–304, 2009. eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1756-8765.2009.01020.x>.
- [20] L. Gisslen, A. Eakins, C. Gordillo, J. Bergdahl, and K. Tollmar. Adversarial Reinforcement Learning for Procedural Content Generation. In *2021 IEEE Conference on Games (CoG)*, pages 1–8, Copenhagen, Denmark, Aug. 2021. IEEE.
- [21] J. Hare. Dealing with Sparse Rewards in Reinforcement Learning. *arXiv:1910.09281 [cs, stat]*, Nov. 2019. arXiv: 1910.09281.
- [22] L. Haug, S. Tschitschek, and A. Singla. Teaching Inverse Reinforcement Learners via Features and Demonstrations. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.
- [23] D. Hood, S. Lemaignan, and P. Dillenbourg. When Children Teach a Robot to Write: An Autonomous Teachable Humanoid Which Uses Simulated Handwriting. In *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction, HRI ’15*, pages 83–90, New York, NY, USA, Mar. 2015. Association for Computing Machinery.
- [24] R. A. Howard. *Dynamic programming and Markov processes*. Technology Press of Massachusetts Institute of Technology, Cambridge, 1960.
- [25] W. Johal. Research Trends in Social Robots for Learning. *Current Robotics Reports*, 1(3):75–83, Sept. 2020.
- [26] P. Kamalaruban, R. Devidze, V. Cevher, and A. Singla. Interactive Teaching Algorithms for Inverse Reinforcement Learning. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, pages 2692–2700, Macao, China, Aug. 2019. International Joint Conferences on Artificial Intelligence Organization.
- [27] B. Kartal, N. Sohre, and S. J. Guy. Data driven sokoban puzzle generation with monte carlo tree search. In *Twelfth Artificial Intelligence and Interactive Digital Entertainment Conference*, 2016.
- [28] A. Khalifa, P. Bontrager, S. Earle, and J. Togelius. PCGRL: procedural content generation via reinforcement learning. In *Proceedings of the Sixteenth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, AIIDE’20*, pages 95–101. AAAI Press, Oct. 2020.
- [29] B. R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. A. A. Sallab, S. Yogamani, and P. Pérez. Deep Reinforcement Learning for Autonomous Driving: A

- Survey. *IEEE Transactions on Intelligent Transportation Systems*, pages 1–18, 2021. Conference Name: IEEE Transactions on Intelligent Transportation Systems.
- [30] E. Konijn and J. Hoorn. Robot tutor and pupils’ educational ability: Teaching the times tables. *Computers & Education*, 157:103970, July 2020.
- [31] M. Lanz, R. Pieters, and R. Ghabcheloo. Learning environment for robotics education and industry-academia collaboration. *Procedia Manufacturing*, 31:79–84, Jan. 2019.
- [32] J. Li. The benefit of being physically present: A survey of experimental works comparing copresent robots, telepresent robots and virtual agents. *International Journal of Human-Computer Studies*, 77:23–37, May 2015.
- [33] R. V. Lindsey, M. C. Mozer, W. J. Huggins, and H. Pashler. Optimizing Instructional Policies. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.
- [34] C. Liu, R. Fang, and J. Chai. Towards Mediating Shared Perceptual Basis in Situated Dialogue. page 10.
- [35] N. G. Lobczowski. Bridging gaps and moving forward: Building a new model for socioemotional formation and regulation. *Educational Psychologist*, 55(2):53–68, Apr. 2020. Publisher: Routledge eprint: <https://doi.org/10.1080/00461520.2019.1670064>.
- [36] N. Lubold, E. Walker, H. Pon-Barry, Y. Flores, and A. Ogan. Using Iterative Design to Create Efficacy-Building Social Experiences with a Teachable Robot. July 2018. Publisher: International Society of the Learning Sciences, Inc. [ISLS].
- [37] S. Matsuzoe, H. Kuzuoka, and F. Tanaka. Learning English words with the aid of an autonomous care-receiving robot in a children’s group activity. *The 23rd IEEE International Symposium on Robot and Human Interactive Communication*, 2014.
- [38] D. Miller, I. Nourbakhsh, and R. Siegwart. Robots for Education. In *Springer Handbook of Robotics*. Jan. 2008. Journal Abbreviation: Springer Handbook of Robotics.
- [39] R. Mitnik, M. Nussbaum, and A. Soto. An autonomous educational mobile robot mediator. *Autonomous Robots*, 25:367, Nov. 2008.
- [40] D. P. Newton and L. D. Newton. Humanoid Robots as Teachers and a Proposed Code of Practice. *Frontiers in Education*, 4, 2019.
- [41] T. J. Nokes-Malach, J. E. Richey, and S. Gadgil. When is it better to learn together? Insights from research on collaborative learning. *Educational Psychology Review*, 27(4):645–656, 2015. Place: Germany Publisher: Springer.
- [42] OpenAI, I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, J. Schneider, N. Tezak, J. Tworek, P. Welinder, L. Weng, Q. Yuan, W. Zaremba, and L. Zhang. Solving Rubik’s Cube with a Robot Hand. Oct. 2019.
- [43] S. B. F. Paletz and C. D. Schunn. Assessing group-level participation in fluid teams: Testing a new metric. *Behavior Research Methods*, 43(2):522–536, June 2011.
- [44] H. W. Park, I. Grover, S. Spaulding, L. Gomez, and C. Breazeal. A Model-Free Affective Reinforcement Learning Approach to Personalization of an Autonomous Social Robot Companion for Early Literacy Education. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:687–694, July 2019.
- [45] T. Patikorn and N. T. Heffernan. Effectiveness of Crowd-Sourcing On-Demand Assistance from Teachers in Online Learning Platforms. In *Proceedings of the Seventh ACM Conference on Learning @ Scale, L@S ’20*, pages 115–124, New York, NY, USA, Aug. 2020. Association for Computing Machinery.
- [46] A. Powers, S. Kiesler, S. Fussell, and C. Torrey. Comparing a computer agent with a humanoid robot. In *Proceedings of the ACM/IEEE international conference on Human-robot interaction*, pages 145–152, 2007.
- [47] A. Rafferty, H. Ying, and J. Williams. Statistical Consequences of using Multi-armed Bandits to Conduct Adaptive Educational Experiments. *Journal of Educational Data Mining*, 11(1):47–79, June 2019. Number: 1.
- [48] A. N. Rafferty, E. Brunskill, T. L. Griffiths, and P. Shafto. Faster Teaching via POMDP Planning. *Cognitive Science*, 40(6):1290–1332, Aug. 2016.
- [49] A. N. Rafferty, H. Ying, and J. J. Williams. Bandit assignment for educational experiments: Benefits to students versus statistical power. In *International Conference on Artificial Intelligence in Education*, pages 286–290. Springer, 2018.
- [50] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann. Stable-Baselines3: Reliable Reinforcement Learning Implementations.
- [51] M. Saerbeck, T. Schut, C. Bartneck, and M. Janse. Expressive Robots in Education Varying the Degree of Social Supportive Behavior of a Robotic Tutor. volume 3, pages 1613–1622, Jan. 2010.
- [52] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal Policy Optimization Algorithms. *arXiv:1707.06347 [cs]*, Aug. 2017. arXiv: 1707.06347.
- [53] A. Segal, Y. B. David, J. J. Williams, Y. Gal, and Y. Shalom. Combining Difficulty Ranking with Multi-Armed Bandits to Sequence Educational Content. In *AIED*, 2018.
- [54] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, Jan. 2016. Number: 7587 Publisher: Nature Publishing Group.
- [55] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. Lillicrap, K. Simonyan, and D. Hassabis. Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm. *arXiv:1712.01815 [cs]*, Dec. 2017. arXiv: 1712.01815.
- [56] A. Singla, A. N. Rafferty, G. Radanovic, and N. T. Heffernan. Reinforcement Learning for Education:

- Opportunities and Challenges. July 2021.
- [57] A. Soller. Supporting Social Interaction in an Intelligent Collaborative Learning System. page 24.
 - [58] F. Tanaka, K. Isshiki, F. Takahashi, M. Uekusa, R. Sei, and K. Hayashi. Pepper learns together with children: Development of an educational application. *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, 2015.
 - [59] F. Tanaka and S. Matsuzoe. Children teach a care-receiving robot to promote their learning. *HRI 2012*, 2012.
 - [60] S. Tschiatschek, A. Ghosh, L. Haug, R. Devidze, and A. Singla. Learner-aware Teaching: Inverse Reinforcement Learning with Preferences and Constraints. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.
 - [61] P. Van den Bossche, W. Gijsselaers, M. Segers, G. Woltjer, and P. Kirschner. Team learning: building shared mental models. *Instructional Science*, 39(3):283–301, May 2011.
 - [62] J. J. Williams, J. Kim, A. Rafferty, S. Maldonado, K. Z. Gajos, W. S. Lasecki, and N. Heffernan. AXIS: Generating Explanations at Scale with Learnersourcing and Machine Learning. In *Proceedings of the Third (2016) ACM Conference on Learning @ Scale, L@S '16*, pages 379–388, New York, NY, USA, Apr. 2016. Association for Computing Machinery.
 - [63] J. J. Williams, A. N. Rafferty, D. Tingley, A. Ang, W. S. Lasecki, and J. Kim. Enhancing Online Problems Through Instructor-Centered Tools for Randomized Experiments. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pages 1–12. Association for Computing Machinery, New York, NY, USA, Apr. 2018.
 - [64] W. Zhao, J. P. Queralta, and T. Westerlund. Sim-to-Real Transfer in Deep Reinforcement Learning for Robotics: a Survey. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 737–744, Dec. 2020.

Detecting SMART Model Cognitive Operations in Mathematical Problem-Solving Process

Jiayi Zhang¹, Juliana Ma. Alexandra L. Andres¹, Stephen Hutt¹, Ryan S. Baker¹,
Jaclyn Ocumpaugh¹, Caitlin Mills², Jamiella Brooks¹, Sheela Sethuraman³, Tyron Young⁴

¹University of Pennsylvania

²University of New Hampshire

³CueThink

⁴Oakwood School

joycez@upenn.edu

ABSTRACT

Self-regulated learning (SRL) is a critical component of mathematics problem solving. Students skilled in SRL are more likely to effectively set goals, search for information, and direct their attention and cognitive process so that they align their efforts with their objectives. An influential framework for SRL, the SMART model, proposes that five cognitive operations (i.e., searching, monitoring, assembling, rehearsing, and translating) play a key role in SRL. However, these categories encompass a wide range of behaviors, making measurement challenging – often involving observing individual students and recording their think-aloud activities or asking students to complete labor-intensive tagging activities as they work. In the current study, we develop machine-learned indicators of SMART operations, in order to achieve better scalability than other measurement approaches. We analyzed student’s textual responses and interaction data collected from a mathematical learning platform where students are asked to thoroughly explain their solutions and are scaffolded in communicating their problem-solving process to their peers and teachers. We built detectors of four indicators of SMART operations (namely, assembling and translating operations). Our detectors are found to be reliable and generalizable, with AUC ROCs ranging from .76-.89. When applied to the full test set, the detectors are robust against algorithmic bias, performing well across different student populations.

Keywords

self-regulated learning, SMART model, automated detectors

1. INTRODUCTION

Work over the last two decades has developed automated detectors of a range of behaviors and constructs in students’ interaction with computer-based learning environments [12]. These detectors utilize log data collected from these learning environments to infer the presence or absence of a complex behavior or a construct in student learning. For example, detectors have been built to identify student affect (e.g., [7, 21, 31]), engagement (e.g., [6, 37]), and problem-solving strategies [43]. Such detectors can be split into two broad categories: detectors for post-hoc analysis and detectors for real-time adaptation. Post-hoc analysis allows researchers to detect

constructs retrospectively and subsequently understand their prevalence (e.g., [32]) and conduct further analysis (e.g., [17, 41]). Detectors designed to be run in real-time facilitate adaptive experiences and real-time feedback [48], as well as reports to teachers [1].

In particular, considerable work has been devoted to detecting and understanding behaviors and strategies involved in self-regulated learning (SRL). By examining student behavior patterns, automated detectors have been developed for a range of SRL related constructs, including help avoidance [3], gaming the system [7], setting goals [5, 14], and planning and tracking progress [14]. However, the specific constructs being modeled often have not been clearly linked to any of the growing number of theoretical models of SRL (although [29] is an exception to this) and have mostly been operationalized in terms of high-level strategies that combine several of behaviors treated as separate in SRL theories, rather than the finer-grained behaviors used in those theories [cf. 46, 48]. Capturing fine-grained indicators of key aspects of SRL in terms of these theoretical models may yield a better understanding of the process of SRL and help EDM research make more direct theoretical contributions.

Self-regulation is a critical component of learning, and has been positively associated with learning outcomes [20, 34, 57]. In mathematics problem-solving, students who are skilled in SRL are able to effectively set goals, search for information, and direct their attention and cognitive resources to align their efforts with their objectives [56]. As a result, SRL facilitates the successful problem-solving process [20, 34, 57] and enables students to acquire a deep and conceptual understanding of the embedded knowledge [31]. Given its benefits, theory-based interventions have been developed to promote SRL [21]. However, current SRL assessments, such as self-reports and think-aloud activities, are not sufficient to provide measurement at scale; at the same time, existing scalable SRL assessments based on automated detection in log data are typically not connected back to theory, making it difficult to use them in theory-driven interventions. SRL assessments based on automated detectors have therefore been used in more ad-hoc, system-specific interventions, often with unintended consequences or unexpected patterns of findings [2, 35]

In the current study, we develop automated detectors that identify fine-grained evidence of SRL constructs drawn from theory. This study does so in the context of CueThink, a digital learning application that focuses on enhancing middle school student mathematics problem-solving skills. Through the lens of the SMART model of SRL (described in greater detail below) [53], we identify and operationalize five SRL indicators: numerical representation, contextual representation, strategy orientation, outcome orientation, and data transformation. We then build automated

J. Zhang, J. M. A. L. Andres, S. Hutt, R. S. Baker, J. Ocumpaugh, C. Mills, J. Brooks, S. Sethuraman, and T. Young. Detecting SMART model cognitive operations in mathematical problem-solving process. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 75–85, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853161>

detectors for each indicator, evaluate their performance, and check them for algorithmic bias.

2. BACKGROUND

2.1 SRL and the SMART Model

Grounded in information processing theory, Winne and Hadwin [55] characterize the process of SRL as four interdependent and recursive stages, in which learners: 1) define the task, 2) set goals and form plans, 3) enact the plans, and 4) reflect and adapt strategies when goals are not met.

The SMART model of SRL [53] was later proposed to further elucidate the processes involved in these four tasks. Specifically, the model separates the “cognitive and behavioral actions applied to perform the task” into five categories: *searching*, *monitoring*, *assembling*, *rehearsing*, and *translating*. Each operation describes a way that learners cognitively engage and interact with information. For example, when working on a task, learners direct their attention to particular information (searching) and compare the information with a standard (monitoring), evaluating the relevance or the importance of the information. When relevant information is identified, students relate pieces of information to one another (assembling), in order to create a comprehensive understanding of the problem. When information does not fit into the current problem representation, learners manipulate the ways information is presented in order to find a solution (translating). Throughout the process, working memory is used to actively maintain and reinstate information (rehearsing).

These cognitive operations are an integral aspect of self-regulation: they help determine student success at completing each of the four SRL tasks, which, in turn, influences the progression of the problem-solving process [52]. However, despite the SMART categories' importance for SRL, they are often difficult to observe or measure, as most learning activities (whether online or offline) do not fully reify the cognitive process involved in their learning tasks. Further, these operations may occur non-linearly, and multiple operations can be employed when completing the same task—making the measurement of these constructs challenging.

2.2 Challenges in SRL Measurements

SRL has typically been measured using three common approaches: self-reports, think-aloud activities, and log data collected in computer-based learning environments [51]. With traditional self-report studies, students are asked about their SRL process either outside of a task (i.e., before or after completing a task) or while working on a task. In decontextualized self-report (outside of a task), students report on the SRL strategies they plan to use or recall on the strategies they used, using a pre- or post-task survey. Even though this approach is widely used, the nature of surveying cognitive processes outside of the task may lead to inaccuracies in the representation of cognition [51]. For example, when recalling a cognitive process retrospectively, students may aggregate the out-of-context, self-reported experience across numerous tasks, failing to demonstrate the relationship between the task and the corresponding SRL strategies. For this reason, several studies have adopted in-context self-report (e.g., [42, 47, 54]), in which students are asked to tag their SRL strategies as they occur.

Other research has leveraged think-aloud activities that ask students to verbalize their cognitive processes when solving a problem [28]. As with in-context self-reports, think-alouds give researchers an opportunity to identify processes that are contextualized in the problem-solving activity and are approximately concurrent with

their occurrences. However, this process can suffer from an observation effect. Students being prompted to discuss their thinking process in real-time may alter that process and not provide an accurate representation of the processes they would engage in naturally [16, 44]. This, in turn, calls into question the validity of findings obtained using this type of measurement and whether they are generalizable to new students and contexts.

2.3 Use Log Data to Measure SRL in Computer-Based Learning Environments

Both self-report and think-aloud approaches are labor-intensive and time-consuming, which make them difficult to scale. As such, a third approach, analyzing log data collected from computer-based learning environments, has emerged as a promising way to measure SRL.

Aleven and colleagues [3] designed an exhaustive set of production rules to represent help-seeking behaviors within a geometry learning system, and then compared these rules to student problem-solving steps to determine whether those steps were warranted by the current situation. Similarly, Biswas et al. [14] used sequences of student behaviors to model a range of SRL behaviors, such as monitoring through explanation, self-assessment, tracking progress, and setting learning goals. Additionally, Segedy et al. [45] utilized log data and coherence analysis to assess students' ability to seek out, interpret, and apply information in an open-ended learning environment, examining if a student's subsequent action is coherent based on the information presented.

Researchers have also used textual responses within dialogue-based learning systems to measure SRL. Graesser and colleagues [27] used latent semantic analysis to study student conversations with animated pedagogical agents to assess and support SRL. Students who frequently use questions in a conversation can be interpreted as showing initiative, and engagement in monitoring can be inferred when students demonstrate in their responses that they feel they know the answer [26].

However, often the log data collected does not straightforwardly reflect a SRL construct [4]. Researchers must decide what data to use, what constructs to measure, and how to operationalize the constructs with the existing data [30]. SRL, as a process, covers a range of behaviors and strategies, so the constructs can vary depending on how SRL is conceptualized and also based on the design of the activity the learner is participating in. To ensure the validity of the operationalization, it is recommended that the operationalization should be conceptualized in terms of a SRL model and contextualized in the learning environment where the data is generated [51].

2.4 Algorithmic Bias

In order to use detectors at scale, we must ensure that they will be valid for the entire populations they are scaled to rather than only subgroups of students. Recent evidence suggests that many published detectors are prone to algorithmic bias, functioning better for some populations of learners than others [11]. However, there has been limited attention to algorithmic bias within the field of educational data mining, where analyses of algorithmic bias are rare and even overall population demographics are only reported in 15% of publications [38]. Verifying detector fairness is particularly pertinent to our study given the diverse student population who use CueThink, making it important to evaluate detector effectiveness across demographic groups before deploying and using the detectors at scale.

2.5 Current Study

In the current study, we address the challenges noted above by building automated detectors of SRL constructs from a theory-driven lens. Using a dataset of 79 students as they interacted with the online math learning platform CueThink, we first examine the learning environment understanding how students interact with the platform and the context of where the log data is generated. Based on the context and the log data available, we identify relevant theoretical constructs grounded in the SMART model. In particular, five SRL indicators relating to cognitive operations in the SMART model (in this case, all either *assembling* or *translating*) are identified for investigation.

We use text replay to code student interactions for each defined indicator. These labels are then used as ground truth for machine learning. We distill a variety of features from the log data to represent multiple aspects of a student’s interaction, including the number of responses and the content in the responses. The ground truth and the features are then input into a machine learning process, training a model to emulate human coders’ judgement, making predictions on the presence or absence of a SRL indicator.

We demonstrate that trained detectors provide accurate detection, suitable for real-time use. Finally, we also evaluate, through slicing analysis, the performance of our models across different demographic groups.

3. METHODS

3.1 Learning Environment

CueThink is a digital learning application that focuses on enhancing middle school student math problem-solving skills, by encouraging students to engage in self-regulated learning and to develop math language to communicate problem-solving processes. CueThink asks students both to solve a math problem and to create a shareable screen-cast video that provides the student’s answer and also demonstrates their problem-solving process. As Figure 1 shows, CueThink structures a problem into a *Thinklet*, a process that includes four phases—*Understand*, *Plan*, *Solve*, and *Review*—that closely align with Winne & Hadwin’s model of SRL [55].

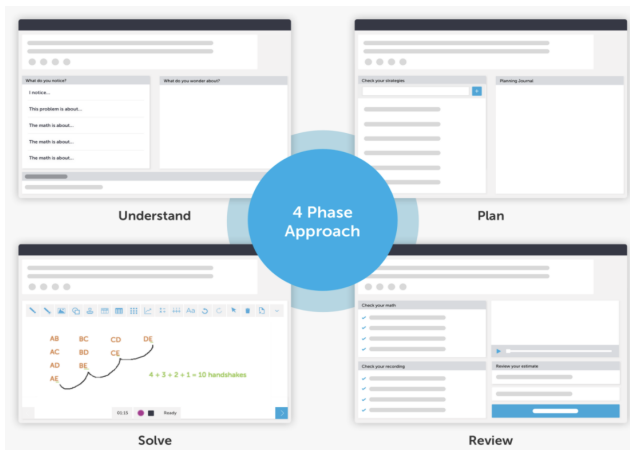


Figure 1. Screenshots of CueThink’s 4 Phase Approach

Each phase of the *Thinklet* (outlined in Table 1 and described in more detail below), asks students to focus on a different part of the problem-solving process. While working on a *Thinklet*, students can move freely across the four phases, including going back to a previous phase or skipping phases.

Starting with the *Understand* phase, students read a problem and provide text-based responses to three questions: (1) “What do you notice?” (2) “What do you wonder?” and (3) “What is your estimated answer to the problem?” This phase encourages students to actively look for information in the problem and create a representation of the problem space. Thus, students demonstrate their understanding of what they know and what they need to know at this phase.

In the *Plan* phase, students build on what they have established in the *Understand* phase by planning how they will solve the problem. Students are first prompted to select what strategies they will use to solve the problem. They may choose from a predefined strategy list (i.e., draw a picture, model with an equation, work backwards, etc.) or define their own strategies. Once the student has selected which strategies they will use, the student is prompted to write a plan on how they will use the strategies to solve the problem.

In the *Solve* phase, students explain and present their answer. Specifically, they create a screen cast video using an interface that provides them with a whiteboard and mathematical tools (i.e., number lines, ruler, etc.).

In the *Review* phase, students provide the final answer to the math problem, but also reflect on whether the answer makes sense and whether their communication is clear, using checklists to scaffold their reflection.

Once students have completed the problem, they share their screen-cast explanation for *Peer Review*. In this phase, teachers and peers annotate both the textual responses and video, often asking the student for their underlying reasoning or why the student picked specific methods. These annotations are then sent back to the video’s author for possible revision.

Table 1. Summary of Responses by *Thinklet* Phase

Phase	Tasks Description & Data Types
Understand	What I notice is (<i>textual response</i>)
	What I wonder is (<i>textual response</i>)
	Estimate your answer (<i>textual response</i>)
Plan	Choose your strategies (<i>select all that apply, textual response</i>)
	Planning journal (<i>textual response</i>)
Solve*	Video creation tools (<i>Whiteboard, math tools, and recording tools</i>)
Review	Check your math (<i>select all that apply</i>)
	Check your recoding (<i>select all that apply</i>)
	Review your estimate (<i>textual response</i>)
	Final answer (<i>textual response</i>)

* Student activity in the *Solve* phase is not used in this paper’s analyses

3.2 Student Demographics

In this study, 79 students in grade 6 and 7 at a suburban school in the southwestern U.S. used CueThink during the 2020-21 school year. The school contains a diverse student population with around 40% Hispanic or Latino, 40% White, 15% African American, and 5% Asian students. Students’ self-reported demographic information on gender and race/ethnicity was collected. For gender, students could choose to identify as male, female, non-binary, or leave the question blank. For race/ethnicity, options included African America, Hispanic/Latinx, White, Asian, Native American, two or more races, other, or prefer not to say. Students reporting

“other” for their race/ethnicity were provided the option to give detail.

3.3 Log Data

CueThink was used in six classrooms over multiple weeks, with teachers assigning problems for students to complete in the application. We collected log files that reflect how students use the application and their problem-solving process. On average, students spent 5.2 hours in CueThink and 1.8 hours working on each *Thinklet*. Specifically, for each problem, encapsulated in a *Thinklet*, data generated during the problem-solving process that includes the questions students answered and their textual responses at each phase were collected. In this study, we analyzed textual and click-stream data, but did not analyze data from the videos. In total, we collected 349 *Thinklets* from 79 students working on 24 different problems. Of those 349 *Thinklets*, not all were first attempts. Students have the opportunity to revise their work, which creates another *Thinklet*. In those cases, it is possible that students do not go through the entire problem-solving process. Of the total number of *Thinklets*, 146 were duplicate attempts.

4. BUILDING DETECTORS

Building automated detectors of self-regulated behaviors was a multi-step process (detailed in the following subsections). First, we distilled human-readable text replays from log data. Using these text replays, we identified and operationalized qualitative categories that corresponded with SRL constructs, grounding the operationalization in Winne’s SMART model. We then labeled the self-regulated behaviors, generating ground truth data. Feature engineering and feature distillation were conducted and used to train the predictive models. Lastly, we evaluated model performance and checked for algorithmic bias.

4.1 Text Replays of Interaction Logs

To facilitate inspection and exploration of the data, we used text replays. This method presents segments of interaction data (referred to as clips) in a human-readable presentation. This process facilitates both initial exploration of the data (such as in section 4.2) along with the final coding process (section 4.3). Clips are then viewed by human coders who label them accordingly [8]. Previous studies have used text replay coding to label student affect, disengagement, and learning strategies, such as gaming the system [10], confusion [32], player goals [22], and SRL strategies such as whether a student is using a table to plan their analyses [43]. This approach demonstrates a similar level of reliability as classroom observations and is 2-6 times faster compared to other methods of generating labels, such as classroom observations, screen replay, and retrospective think/emote-aloud protocols [8].

The length and the grain-size of text replay clips can vary depending on both the available data and the granularity of the predictions the researcher intends to make. Because this study seeks to detect cognitive SRL operations in the problem-solving process, which requires a comprehensive examination across questions and phases, the log files were delineated into clips on the level of entire *Thinklets*. Each clip contains a student’s actions and text-based responses that were submitted as that student worked through the four phases to produce a single *Thinklet*. (Note that because video data was not available in these replays, our coders did not see information from the *Solve* phase.) The clips were distilled from log files and presented using a Python window, shown in Figure 2. As video data could not be trivially converted into a common text format (due to wide variability in the videos), video data was not included in the text replays at this time. Creating videos is the primary action

in the *Solve* phase, thus the text replays did not contain any data from the *Solve* phase. Coding videos explicitly poses challenges to scalability of the work and is beyond the scope of this paper, but may be considered in future work.

4.2 Construct Operationalization

To identify constructs to detect, we first examined the clips containing student responses in *Thinklets* and coded student responses for indicators of SRL—qualitative categories that correspond with SRL constructs (we discuss the details of exactly how the data was coded in section 4.3). The definitions of the indicators we coded were developed through dialogue between the research team and system developers. This process followed the recursive, iterative process used in [49] that includes seven stages: conceptualization of codes, generation of codes, refinement of the first coding system, generation of the first codebook, continued revision and feedback, coding implementation, and continued revision of the codes [49]. The conceptualization of codes included a review of related literature, including several theoretical frameworks and perspectives [13, 15, 24], primarily focusing on the SMART model [53]. Using grounded theory [18], we identified common behaviors that were (1) indicative of SRL as characterized by Winne’s SMART model [53] and (2) salient in the log files. A draft lexicon and multiple criteria were generated for a coding system to help identify these constructs.

Given the learning environment’s design and the available data, our efforts focused on defining behaviors related to two categories of cognitive operations (the assembling and translating operations from the SMART model) as they are frequently employed in the initial stages of SRL as learners define tasks and set goals. Following the process used in [49], two coders (the first and second authors) coded a set of clips together, identified five SRL indicators (i.e., numerical representation, contextual representation, strategy orientation, outcome orientation, and data transformation), outlined the criteria for each indicator, and created a rubric.

The draft coding manual was discussed with all members of the research team and developers and designers at CueThink to build a common understanding of the criteria and constructs being examined as well as the features of the system to gain feedback for further refinement. This process was repeated until the entire team had reached a shared understanding of the criteria and constructs being examined by the codebook. The SRL indicators identified, the criteria, and alignment with the SMART model are included in Table 2.

Numerical and contextual representation consider a learner’s process of creating a problem representation, which often occurs in the initial stage in the problem-solving process (i.e., define the task), outlined in the four-phase model of SRL [55]. In problem representation, learners create a problem space by identifying information they know and information they need to know. The two SRL indicators encode how learners represent and process information in math problems, denoting if numerical components and/or contextual details are noted. We consider both of these processes to reflect assembling in the SMART model as students are creating their representation of the data from the information provided. There may also be overlap with translating in some cases, especially if the question provides a different representation to the one the students use. However, as this is not always the case, we primarily consider both indicators to reflect assembling actions and tag translating actions in a different code (see below).

Strategy and outcome orientation also reflect student assembling behaviors. Both indicators consider how students set their goals and form plans for the problem-solving process. These two indicators demonstrate a difference in focuses (process vs. output).

Lastly, data transformation reflects behaviors that are associated with the translating operation, in which the learner manipulates the ways information is represented to them in the problem to find a solution.

Table 2. SRL Indicators Coded through Text Replays

SMART Category	SRL Indicator	Working Definition
Assembling	Numerical Representation (NR)	The learner’s representation of the problems includes numerical components and demonstrates a level of understanding of how the numerical values are used in the math problem.
Assembling	Contextual Representation (CR)	The learner’s representation of the problem includes contextual details relating to the setting/characters/situations within the given math problem.
Assembling	Strategy Orientation (SO)	Learners explicitly state a plan for how they will find the answer for the given math problem, decomposing information into a step-by-step process.
Assembling	Outcome Orientation (OO)	The learner provides only a numerical estimate of the final answer for the given math problem, suggesting that learners are focused on the output instead of the process itself.
Translating	Data Transformation (DT)	The learner manipulates the ways information is represented to them in the problem to find a solution. This suggests active problem solving.

4.3 Coding the Data

After constructs were operationalized and defined, we proceeded to code the remainder of the data. Two coders, (the same as in the previous section), completed the text replay coding in three phases: preliminary coding (discussed above), separate coding (two coders per clip; for establishing inter-rater reliability), and individual coding (one coder per clip; for completeness).

The two coders each used the codebook/rubric to code the same set of clips separately. They then compared the labels and computed the inter-rater reliability (IRR) kappa. For constructs with low kappa, the two coders discussed their differences in labeling and conducted another round of coding. This step of separate coding and comparing is repeated until an acceptable reliability is established. After two rounds of coding, as shown in Table 3, the two coders reached an acceptable IRR above 0.60 for all five SRL indicators (M=0.75).

Table 3. Inter-Rater Reliability in Separate Coding

SRL Indicator	IRR Kappa
Numerical Representation (NR)	0.83
Contextual Representation (CR)	0.63
Strategy Orientation (SO)	0.74
Outcome Orientation (OO)	0.78
Data Transformation (DT)	0.74

Once the reliability was established, the coders moved on to the individual coding where they split the rest of the clips and coded them individually.

Each construct was considered over the entire *thinklet*. Thus, in total, the two coders coded 349 clips. However, in order to consistently examine the entire problem-solving process, 167 clips that were marked incomplete because students stopped before completing the entire problem were excluded. Of the remaining 182 clips, coding resulted in the following distribution of labels: 64% numerical representation, 77% contextual representation, 8% strategy orientation, 72% outcome orientation, and 73% data transformation. These were produced by 72 students, who, on average, each contributed 3 clips (max=4, min=1, median=3).

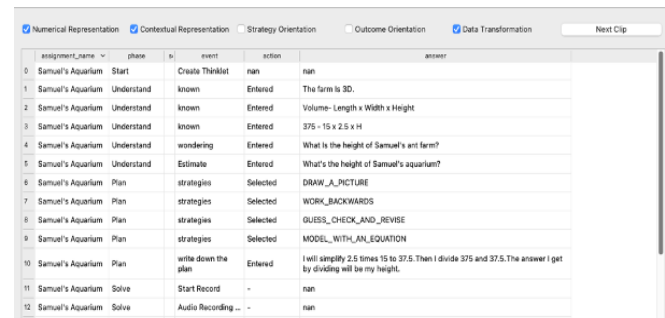


Figure 2. Screenshot of Text Replay Coding Window

4.4 Feature Distillation

Two sets of features were distilled to build the detectors. Both sets of features consist solely of features that can be extracted and used in real-time. The first set of features were designed to provide an overview of a *Thinklet* by examining the number of responses in a *Thinklet*. These features (N = 10) were distilled at the *Thinklet* level. For example, we distilled the number of questions students answered in a *Thinklet* and the number of responses in each phase. To understand the strategies that students select in the *Plan* phase, we also created a feature that counts the number of strategies a student selects among the top two strategies used by peers for the same problem.

The second set of features were designed to examine the content and the linguistic features of students’ text-based responses. These features (N = 90) were first extracted at the response level and then aggregated to the phase level. These aggregations were calculated for the *Understand*, *Plan*, and *Review* phases. (No textual data was extracted from the *Solve* phase as there was no textual input in this phase.)

Specifically, we distilled whether each response: 1) contains a numerical value, 2) consists of only numerical values, 3) has mathematical operation signs, 4) contains a question (if it contains a question mark or uses keywords such as “wonder”, “why”, etc.), 5) uses language that indicates the formation of a plan (e.g., the use of keywords like “plan”, “I will”, “going to”, etc.) , and 6) is the exact repetition of a previous answer. These criteria generate a set

of binary variables for each response. We averaged these binary variables across the responses within a phase, creating 18 features for each *Thinklet*.

Additionally, in each response, we counted the number of 7) characters, 8) words, 9) numerical values, 10) verbs, 11) nouns, and 12) pronouns. Features 10-12 were counted using Udpipes, a natural language processing toolkit [50]. We also 13) counted the number of keywords used from a predefined list that provides the context of each problem; and 14) computed how similar each response is to the problem item using the Smith-Waterman algorithm [46]. For these continuous variables, we computed the mean, standard deviation, and max of the values for each phase, creating 72 features.

Features distilled from the two sets were combined. In total, 100 features were extracted from each *Thinklet* process and were then used to construct the automated detectors. Note that we did not extract any features from the video that students make in the current work. Similarly, we did not use any of the audio from the video (or transcription thereof) for any features.

4.5 Machine Learning Algorithms

We used the scikit-learn library [40] to implement commonly-used models, including Logistic Regression, Lasso, Decision Tree, Random Forest as well as Extreme Gradient Boosting (XGBoost) as implemented in the XGBoost library [19]. XGBoost outperformed other algorithms in all cases; we therefore only discuss the XGBoost results below.

XGBoost uses an ensemble technique that trains an initial, weak decision tree and calculates its prediction errors. It then iteratively trains subsequent decision trees to predict the error of the previous decision tree, with the final prediction representing the sum of the predictions of all the trees in the set. We tested the detectors with 10-fold student-level cross-validation. For this approach, the dataset was split into 10 student-level folds, meaning that in cases where students had multiple *Thinklets*, all of their data would be contained within the same fold and at no time could data from a student be included in both the training and testing set. Nine folds were used to train the model, and the trained model was used to make predictions for the 10th fold. Each fold acted as the test set once. Student-level cross-validation was conducted to verify generalizability to new students.

Models were evaluated using the area under the Receiver Operating Characteristic curve (AUC ROC), which indicates the probability that the model can correctly distinguish between an example of each class. An AUC ROC of 0.5 represents chance classification, while an AUC ROC of 1 represents perfect classification. Results were calculated for each fold and averaged to yield one AUC ROC score per detector.

5. RESULTS

5.1 Model Performance

Due to the rarity of strategy orientation (only 14 clips were labeled with this construct), a detector could not be built for this construct. Automated detectors were built for the other four constructs. As shown in Table 4, the average AUC ROC derived from 10-fold student-level cross-validation is 0.894 for numerical representation (NR), 0.813 for contextual representation (CR), 0.761 for outcome orientation (OO), and 0.815 for data transformation (DT). These findings suggest that the detectors were generally successful at capturing these four SRL constructs. We also calculated the standard deviations (SD) of the AUC ROCs across the 10 folds for each detector to investigate the variability across folds.

Table 4. Detector Performance Measured by AUC ROC

SRL Indicator	AUC ROC (SD)
Numerical Representation	0.894 (.078)
Contextual Representation	0.813 (.132)
Outcome Orientation	0.761 (.076)
Data Transformation	0.815 (.163)

5.2 Feature Importance

To better understand the detectors as well as to inform our understanding of how these features relate to the constructs, the SHapley Additive exPlanations (SHAP) [33] value, which reflects feature importance, was calculated for each feature within each test set.

These values were then averaged across the 10 testing sets and ranked based on their absolute values. Of the 100 features used, Table 5 reports the top five features with the highest absolute SHAP values for each detector. To understand the directionality, we examined the average SHAP values of the features listed. A positive average SHAP value was found for all the features listed (except for one, as indicated in the Contextual Representation section of Table 5). The positive values indicate that the features are positive predictors of the SRL indicators, suggesting that the higher the values in each feature, the more likely the model is to infer the presence of a SRL indicator.

Table 5. The Top Five Features from each Detector

Feature Phase	Feature
Numerical Representation	
Understand	Mean N of responses that give numerical values
Understand	Max value of the similarity feature which indicates how parallel a student's response is to the original problem
Understand	SD of the similarity feature
Understand	Total N of responses
Plan	Avg value of the similarity feature
Contextual Representation	
Understand	Is there a response to the "what do you notice" question?
Understand	Avg N of keywords used
Understand	SD of the N of characters used (Contextual Representation less likely)
Thinklet	Total N of responses
Plan	Max value of the N of characters used
Outcome Orientation	
Review	Avg N of keywords used
Review	Avg N of words used
Understand	SD of the N of numerical values used
Understand	Is there a response to the "what is your estimated answer" question?
Review	Avg N of nouns used
Data Transformation	
Plan	N of strategies selected that were among the most common strategies used by peers
Understand	SD of the similarity feature
Plan	SD of the N of characters used
Understand	SD of the N of nouns used
Plan	Max value of N of words used

We note that of the 20 features listed in Table 5, 11 are from the *Understand* phase, 5 are from the *Plan* Phase and 3 are from the *Review* phase. In other words, behaviors in the early phases

Table 6. Detector Performance by Gender and Racial/Ethnic Groups

	All Students (k-fold)	All Students (Pooled)	Gender			Race/Ethnicity					
			Male	Female	Left Blank	African American	Hispanic/Latinx	White	Prefer Not to Say	Other	Two or more races
N. students	72	72	33	28	9	6	18	8	19	14	5
N. clips	182	182	81	73	24	20	38	19	50	37	12
NR	0.89	0.89	0.82	0.93	0.97	0.92	0.88	0.96	0.86	0.85	0.86
CR	0.81	0.80	0.74	0.75	0.94	0.75	0.72	0.80	0.90	0.65	0.78
OO	0.78	0.75	0.78	0.74	0.72	0.71	0.81	0.80	0.74	0.78	0.46
DT	0.82	0.86	0.88	0.87	0.78	0.92	0.91	0.83	0.84	0.82	0.86

contributed more heavily to the predictions. This finding aligns with how the *Thinklets* were initially coded. Specifically, the coders primarily examined student responses in the *Understand* phase for numerical and contextual representation as this phase contains information demonstrating how student assemble information and create a problem representation; the coders examined the *Thinklet* more broadly when coding for other SRL indicators, as they encompass behaviors that span across phases.

5.3 Algorithmic Bias

Algorithmic bias occurs when model performance is substantially better or worse across mutually exclusive groups separated by generally non-malleable factors [11]. To validate our detectors, we tested the model performance in different student populations, based on gender and race/ethnicity using slicing analysis [25]. Specifically, utilizing the predictions made in the testing sets, AUC was computed for each subgroup of students in the data for which we received data on group membership. However, due to sample size, comparisons were not possible for gender non-binary students (N=2), Asian students (N=2), or Native American students (N=0).

As Table 6 shows, the difference in model performance measured by AUC between male and female students is small, ranging from 0.01-0.11 for the four detectors. The detectors for numerical representation and contextual representation performed somewhat better for female students ($AUC_{NR} = .93$, $AUC_{CR} = .75$) than for male students ($AUC_{NR} = .82$, $AUC_{CR} = .74$), while detectors for outcome orientation and data transformation performed somewhat better for male students ($AUC_{OO} = .78$, $AUC_{DT} = .88$) than for female students ($AUC_{OO} = .74$, $AUC_{DT} = .87$).

Table 6 also shows the analysis of algorithmic bias in terms of race/ethnicity, comparing the AUC between student racial/ethnic subgroups that had more than 5 students in our sample: African American, Hispanic/Latinx, and White. Small to moderate differences were observed across the three groups, though the differences were not consistent (i.e., no racial/ethnic group consistently had the best-performing detectors). However, performance remained acceptable for all four detectors across all groups. When detecting numerical representation and contextual representations, the detectors performed somewhat better for White students ($AUC_{NR} = 0.96$, $AUC_{CR} = 0.80$), than for African American ($AUC_{NR} = 0.92$, $AUC_{CR} = 0.75$) and Hispanic/Latinx ($AUC_{NR} = 0.88$, $AUC_{CR} = 0.72$) students. However, the outcome orientation detector had somewhat higher performance for Hispanic/Latinx students ($AUC_{OO} = 0.81$), than for White ($AUC_{OO} = 0.80$) and African American ($AUC_{OO} = 0.71$) students. The data transformation detector performed better for African American students ($AUC_{DT} = 0.92$) than for Hispanic/Latinx ($AUC_{DT} = 0.91$) and White ($AUC_{DT} = 0.83$) students.

Performance was substantially lower for two constructs/group combinations: detecting contextual representation for students who identify race as other ($AUC_{CR} = 0.65$) and detecting outcome orientation for students who identify as belonging to two or more races ($AUC_{OO} = 0.46$). These more substantial differences may be due to the small sample size of these constructs within these subgroups; in future work, larger samples will be collected in order to validate performance in these groups.

Given the data collected, we noticed a considerable number of students who declined to report gender (N = 9) and race (N = 19). Both groups performed close to the average model performance, across groups and contexts.

6. CONCLUSION AND DISCUSSION

6.1 Main Findings

Given the importance of self-regulation in learning, specifically in the problem-solving process, an increasing number of studies have looked into ways to promote self-regulated learning. This first requires the ability to accurately measure SRL, so that interventions can be introduced to encourage and guide students to self-regulate effectively. However, the most common ways of measuring SRL in a fine-grained fashion – either through self-report and think-aloud protocols – are difficult to automate and scale, and they can also interrupt or interfere with the learning task. Log data collected from computer-based learning environments offers an unobtrusive and potentially scalable solution to help understand when and how students self-regulate within the problem-solving process, in order to inform decisions on intervention (e.g., [3]). However, previous automated detection of SRL constructs using log data has mostly not been explicitly connected to SRL theory. In the current work, we explored the possibility of detecting SRL constructs at a fine-grained level, focusing on detecting cognitive operations (i.e., assembling and translating), outlined in the SMART model [53]. Specifically, we detected the presence of four self-regulation indicators related to two categories of operations: numerical representation, contextual representation, outcome orientation, and data transformation. Evaluated using 10-fold student-level cross-validation, our detectors were found to be accurate and valid across demographic groups, with AUC ROC ranging from .76-.89.

To understand the detectors, feature importance was examined using SHAP values. The top five features with the highest absolute SHAP values were identified for each detector. With the features identified, we find that except for outcome orientation, the detectors primarily rely on features extracted from the *Understand* and *Plan* phases of the learning activity, the two phases where students assemble information and make plans. In particular, the numerical representation detector mainly relies on features that examine the

numerical values used in the *Understand* phase as well as features that compare the similarity between student responses and the problem item. The numerical value feature makes sense, as the detector is operationalized to identify if numerical components are processed and represented when students assemble information.

However, the maximum similarity feature, a feature that takes both numerical values and text into account, also contributes to the NR indicator. This finding suggests that the NR detector not only examines if numbers are used in responses, but also how they are used in relation to the problem. As such, this finding validates the operationalization of this indicator, showing that the learner demonstrates a level of understanding of how numerical values are used in math problems, creating a representation of the problem space utilizing numbers.

The contextual representation detector looks at the keywords used in student responses in the *Understand* phase and the length of the responses in the *Plan* phase, which indicates the relationship that the longer the responses are when a student is forming a plan, the more likely it is for the student to contextually representing the problem. When predicting the presence of outcome orientation, the model utilizes features extracted in the *Understand* and the *Review* phases, understanding students' use of keywords, nouns, and numerical values in these two phases. At last, the data transformation detector checks the number of top strategies students select as well as the length and the variation in the length of the responses in the *Understand* and *Plan* phase.

Additionally, we examined model performance on different demographic subgroups of students, both in terms of gender and racial/ethnic groups, to verify their fairness and lack of algorithmic biases. Relatively small differences were observed in each comparison, and no student group (either gender or racial/ethnic group) consistently had the best-performing detectors.

6.2 Applications

The detectors built in the current study provide two advantages over previous SRL detectors. First, previous SRL detectors generally identify higher-level strategies and are not typically linked to theory; in contrast, we specifically based our detectors on a SRL model in order to identify theoretically-grounded SRL constructs at a finer-grain size. Having developed these fine-grained models of behavior associated with the assembling and translating operations of SMART, we can conduct analyses to further our understanding of the role that cognitive operations play in the broader process of SRL. For example, we can investigate questions about how often students use these cognitive operations in each of the four tasks outlined in the Winne and Hadwin's four-stage model, and how the engagement and the frequency of the engagement in these cognitive operations contributes to the success of completing the tasks. Results from future analyses will help expand the current theoretical understanding on SRL, adding specificity to the still high-level processes represented in contemporary SRL theory.

Second, given that most previous detectors are not connected back to SRL theory, it has been difficult to use them with theory-driven interventions. The detectors proposed in the current study are developed based on a theoretical model of SRL [46, 48] and are operationalized to capture key aspects of the cognitive operations in the model. These detectors can therefore be used to facilitate the development of adaptive learning environments that respond to student SRL, in a fashion connected to theory. For instance, a student demonstrating an outcome orientation could be encouraged to reflect further on their strategy.

Similarly, these detectors could also provide theory-grounded information to teachers (e.g., through a dashboard), providing insight on how students are approaching problems. This data can inform teachers as they create and refine their problems, as well as informing how they support their students. As with any application of this nature, careful attention will be needed in design to ensure that data is presented in the most useful form for teachers and appropriately represents the uncertainty in the model (i.e., false positives or false negatives).

6.3 Limitations and Future Work

This work has five principal limitations that should be addressed in future work. First, when validating the fairness of the models, the sample size is small (less than five students) for several student groups. Reliable comparison of the model performance for these groups of students is therefore not possible. In future work, larger and more representative samples should be collected in order to validate model performance for a broader range of student groups.

Second, although our detectors are based on a theoretical model of SRL, the operationalization of our constructs is contextualized in the current learning environment, so our detectors may be platform-specific. Future work should study the transferability of the current detectors by examining their applicability and predictive performance, and explore how they can be adapted for use in other learning environments. To the extent that some of our detectors (such as the data transformation detector) apply across learning environments, we can investigate their performance within those contexts to evaluate their transferability (see, for instance, [39]).

Third, since the detectors are currently trained on complete *Thinklets*, they will have some limitations in the ways they can be used when being implemented in a learning platform. Specifically, the detectors will only be able to make predictions after a student has solved a problem, providing an indicator at that point on the student's use or lack of use of assembling and translating in the problem-solving process. As such, these detectors will not provide immediate detection of these strategies when students are working through a problem. However, they can still be used to inform teachers and direct their feedback after a problem has concluded, in between problems or for the next problem. To enable other uses, it may be relevant to examine ways of also making early predictions based on incomplete *Thinklets* in order to provide detection during the problem-solving process, enabling real-time interventions.

Future work should also consider additional methods for ground truth labelling. In this work we used a post-hoc tagging approach (through text replays), to identify indicators of SRL-related strategies. This approach has the potential to miss crucial "in-the-moment" events that are not evident from the log data alone. Future studies could examine how post-hoc tagging used in the current study align with in-the-moment tagging, reported by either student themselves or external observers/interviewers (e.g. [7]) to examine additional aspects of SRL.

Finally, future work should consider expanding the scope of this work. In the current study, five constructs were identified and four modeled. SRL as a process covers a much broader range of behaviors and strategies that elicit the use of various cognitive operations. Future studies should model and detect a broader range of cognitive operations throughout the four stages of self-regulated learning in the context of problem-solving.

6.4 Conclusions

To better understand and facilitate the use of self-regulation in problem-solving, the current study tested the possibility of scaling up SRL measurement by leveraging machine learning to automatically detect individual SRL indicators through the lens of the SMART model. We built automated detectors that identify four commonly used strategies in math problem solving, indicating assembling and translating operations. Our detectors were found to be reliable and generalizable. Additionally, the detectors were also tested on different student populations to verify their fairness and lack of algorithmic bias, addressing a previously overlooked issue in the field of educational data mining. Given these properties, we anticipate implementing the detectors in the learning environment to collect more fine-grained data and to leverage the detection to inform interventions, creating more positive experiences in mathematical problem-solving.

7. ACKNOWLEDGEMENTS

The research reported here was supported by the EF+Math Program of the Advanced Education Research and Development Program (AERDF) through funds provided to the University of Pennsylvania, University of New Hampshire, and CueThink. The opinions expressed are those of the authors and do not represent views of the EF+Math Program or AERDF.

8. REFERENCES

- [1] Aguilar, S.J. et al. 2021. Associations between learning analytics dashboard exposure and motivation and self-regulated learning. *Computers & Education*. 162, (2021). DOI:<https://doi.org/10.1016/j.compedu.2020.104085>.
- [2] Alevin, V. et al. 2016. Help Helps, But Only So Much: Research on Help Seeking with Intelligent Tutoring Systems. *International Journal of Artificial Intelligence in Education*. 26, 1 (2016), 205–223. DOI:<https://doi.org/10.1007/s40593-015-0089-1>.
- [3] Alevin, V. et al. 2006. Toward Meta-cognitive Tutoring: A Model of Help Seeking with a Cognitive Tutor. *International Journal of Artificial Intelligence in Education*. 16, 2 (2006), 101–128.
- [4] Azevedo, R. et al. 2017. Understanding and Reasoning about Real-Time Cognitive, Affective, and Metacognitive Processes to Foster Self-Regulation with Advanced Learning Technologies. *Handbook of Self-Regulation of Learning and Performance*. (2017), 254–270. DOI:<https://doi.org/10.4324/9781315697048-17>.
- [5] Azevedo, R. et al. 2011. Use of hypermedia to assess and convey self-regulated learning. *Handbook of Self-Regulation of Learning and Performance*. 32, (2011), 102–121. DOI:<https://doi.org/10.4324/9780203839010.ch7>.
- [6] Baker, R.S. et al. 2010. Detecting Gaming the System in Constraint-Based Tutors. *Proceedings of the 18th Annual Conference on User Modeling, Adaptation, and Personalization*. (2010), 267–278. DOI:https://doi.org/10.1007/978-3-642-13470-8_25.
- [7] Baker, R.S. et al. 2004. Detecting student misuse of intelligent tutoring systems. *International Conference on Intelligent Tutoring Systems*. (2004), 531–540. DOI:https://doi.org/10.1007/978-3-540-30139-4_50.
- [8] Baker, R.S. et al. 2006. Human Classification of Low-Fidelity Replays of Student Actions. *Proceedings of the Educational Data Mining Workshop at the 8th International Conference on Intelligent Tutoring Systems*. (2006), 29–36.
- [9] Baker, R.S. et al. 2012. Towards Sensor-Free Affect Detection in Cognitive Tutor Algebra. *Proceedings of the 5th International Conference on Educational Data Mining*. (2012), 126–133.
- [10] Baker, R.S. and Carvalho, A.M.J.A. 2008. Labeling student behavior faster and more precisely with text replays. *Proceedings of the 1st International Conference on Educational Data Mining*. (2008), 38–47.
- [11] Baker, R.S. and Hawn, A. 2021. Algorithmic Bias in Education. *International Journal of Artificial Intelligence in Education*. (2021), 1–41. DOI:<https://doi.org/10.35542/osf.io/pbmzv>.
- [12] Baker, R.S. and Yacef, K. 2009. The State of Educational Data Mining in 2009: A Review and Future Visions. *Journal of Educational Data Mining*. 1, 1 (2009), 3–17. DOI:<https://doi.org/10.5281/zenodo.3554657>.
- [13] Bandura, A. 1986. *Social foundations of thought and action: A social cognitive theory*.
- [14] Biswas, G. et al. 2010. Measuring Self-regulated Learning Skills Through Social Interactions in a Teachable Agent. *Research and Practice in Technology Enhanced Learning*. 05, 02 (Jul. 2010), 123–152. DOI:<https://doi.org/10.1142/S1793206810000839>.
- [15] Boekaerts, M. 1999. Self-regulated learning: where we are today. *International Journal of Educational Research*. 31, 6 (1999), 445–457. DOI:[https://doi.org/10.1016/S0883-0355\(99\)00014-2](https://doi.org/10.1016/S0883-0355(99)00014-2).
- [16] Bosch, N. et al. 2021. Students’ Verbalized Metacognition During Computerized Learning. *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (May 2021), 1–12.
- [17] Botelho, A.F. et al. 2018. Studying Affect Dynamics and Chronometry Using Sensor-Free Detectors. *Proceedings of the 11th International Conference on Educational Data Mining*. (2018), 157–166.
- [18] Charmaz, K. 1983. The grounded theory method: An explication and interpretation. *Contemporary Field Research*. (1983), 109–126.
- [19] Chen, T. and Guestrin, C. 2016. XGBoost: A Scalable Tree Boosting System. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (San Francisco California USA, Aug. 2016), 785–794.
- [20] Cleary, T.J. and Chen, P.P. 2009. Self-regulation, motivation, and math achievement in middle school: Variations across grade level and math context. *Journal of School Psychology*. 47, 5 (Oct. 2009), 291–314. DOI:<https://doi.org/10.1016/j.jsp.2009.04.002>.
- [21] Devolder, A. et al. 2012. Supporting self-regulated learning in computer-based learning environments: systematic review of effects of scaffolding in the domain of science education. *Journal of Computer Assisted Learning*. 28, 6 (Dec. 2012), 557–573. DOI:<https://doi.org/10.1111/j.1365-2729.2011.00476.x>.
- [22] DiCerbo, K.E. and Kidwai, K. 2013. Detecting player goals from game log files. *Presented at the 6th International Conference on Educational Data Mining* (2013).
- [23] D’Mello, S.K. et al. 2008. Automatic detection of learner’s affect from conversational cues. *User Modeling and User-Adapted Interaction*. 18, 1–2 (Feb. 2008), 45–80. DOI:<https://doi.org/10.1007/s11257-007-9037-6>.
- [24] Efklides, A. 2011. Interactions of Metacognition with Motivation and Affect in Self-Regulated Learning: The MASRL

- Model. *Educational Psychologist*. 46, 1 (Jan. 2011), 6–25. DOI:<https://doi.org/10.1080/00461520.2011.538645>.
- [25] Gardner, J. et al. 2019. Evaluating the Fairness of Predictive Student Models Through Slicing Analysis. *Proceedings of the 9th International Conference on Learning Analytics & Knowledge* (Tempe AZ USA, Mar. 2019), 225–234.
- [26] Graesser, A. and McNamara, D. 2010. Self-Regulated Learning in Learning Environments with Pedagogical Agents That Interact in Natural Language. *Educational Psychologist*. 45, 4 (Oct. 2010), 234–244. DOI:<https://doi.org/10.1080/00461520.2010.515933>.
- [27] Graesser, A.C. et al. 2007. Using LSA in AutoTutor: Learning through mixed initiative dialogue in natural language. *Handbook of Latent Semantic Analysis*. 243–262.
- [28] Greene, J.A. et al. 2017. Capturing and modeling self-regulated learning using think-aloud protocols. *Handbook of Self-Regulation of Learning and Performance*. 323–337.
- [29] Hutt, S. et al. 2021. Investigating SMART Models of Self-Regulation and their Impact on Learning. *Proceedings of the 14th International Conference on Educational Data Mining*. (2021), 580–587.
- [30] Kovanovic, V. et al. 2016. Does Time-on-task Estimation Matter? Implications on Validity of Learning Analytics Findings. *Journal of Learning Analytics*. 2, 3 (Feb. 2016), 81–110. DOI:<https://doi.org/10.18608/jla.2015.23.6>.
- [31] Labuhn, A.S. et al. 2010. Enhancing students’ self-regulation and mathematics performance: the influence of feedback and self-evaluative standards. *Metacognition and Learning*. 5, 2 (Aug. 2010), 173–194. DOI:<https://doi.org/10.1007/s11409-010-9056-2>.
- [32] Lee, D.M.C. et al. 2011. Exploring the Relationship between Novice Programmer Confusion and Achievement. *International Conference on Affective Computing and Intelligent Interaction*. (2011), 175–184. DOI:https://doi.org/10.1007/978-3-642-24600-5_21.
- [33] Lundberg, S.M. et al. 2019. Consistent Individualized Feature Attribution for Tree Ensembles. *arXiv:1802.03888*. (2019). DOI:<https://doi.org/10.48550/arXiv.1802.03888>.
- [34] Nota, L. et al. 2004. Self-regulation and academic achievement and resilience: A longitudinal study. *International Journal of Educational Research*. 41, 3 (Jan. 2004), 198–215. DOI:<https://doi.org/10.1016/j.ijer.2005.07.001>.
- [35] Ocumpaugh, J. et al. 2021. Using Qualitative Data from Targeted Interviews to Inform Rapid AIED Development. *Proceedings of the 29th International Conference on Computers in Education*. (2021), 69–74.
- [36] Paquette, L. et al. 2014. Sensor-Free Affect Detection for a Simulation-Based Science Inquiry Learning Environment. *International Conference on Intelligent Tutoring Systems*. (2014), 1–10. DOI:https://doi.org/10.1007/978-3-319-07221-0_1.
- [37] Paquette, L. et al. 2014. Towards Understanding Expert Coding of Student Disengagement in Online Learning. *Proceedings of the 36th annual cognitive science conference* (2014), 1126–1131.
- [38] Paquette, L. et al. 2020. Who’s learning? Using demographics in EDM research. *Journal of Educational Data Mining*. 12, 3 (2020), 1–30. DOI:<https://doi.org/10.5281/zenodo.4143612>.
- [39] Paquette, L. and Baker, R.S. 2017. Variations of Gaming Behaviors Across Populations of Students and Across Learning Environments. *Proceedings of the 18th International Conference on Artificial Intelligence in Education*. (2017), 274–286. DOI:https://doi.org/10.1007/978-3-319-61425-0_23.
- [40] Pedregosa, F. et al. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*. 12, (2011), 2825–2830. DOI:<https://doi.org/10.1007/s13398-014-0173-7.2>.
- [41] Richey, J.E. et al. 2021. Gaming and Confrustion Explain Learning Advantages for a Math Digital Learning Game. *Artificial Intelligence in Education*. (2021), 342–355. DOI:https://doi.org/10.1007/978-3-030-78292-4_28.
- [42] Sabourin, J. et al. 2012. Predicting Student Self-regulation Strategies in Game-Based Learning Environments. *Intelligent Tutoring Systems*. (2012), 141–150. DOI:https://doi.org/10.1007/978-3-642-30950-2_19.
- [43] Sao Pedro, M.A. et al. 2013. Leveraging machine-learned detectors of systematic inquiry behavior to estimate and predict transfer of inquiry skill. *User Modeling and User-Adapted Interaction*. 23, 1 (Mar. 2013), 1–39. DOI:<https://doi.org/10.1007/s11257-011-9101-0>.
- [44] Schooler, J.W. et al. 1993. Thoughts Beyond Words: When Language Overshadows Insight. *Journal of Experimental Psychology: General*. 122, 2 (1993), 166–183. DOI:<https://doi.org/10.1037/0096-3445.122.2.166>.
- [45] Segedy, J.R. et al. 2015. Using Coherence Analysis to Characterize Self-Regulated Learning Behaviours in Open-Ended Learning Environments. *Journal of Learning Analytics*. 2, 1 (May 2015), 13–48. DOI:<https://doi.org/10.18608/jla.2015.21.3>.
- [46] Smith, T.F. and Waterman, M.S. 1981. Identification of common molecular subsequences. *Journal of Molecular Biology*. 147, 1 (Mar. 1981), 195–197. DOI:[https://doi.org/10.1016/0022-2836\(81\)90087-5](https://doi.org/10.1016/0022-2836(81)90087-5).
- [47] Taub, M. et al. 2014. Can the use of cognitive and metacognitive self-regulated learning strategies be predicted by learners’ levels of prior knowledge in hypermedia-learning environments? *Computers in Human Behavior*. 39, (Oct. 2014), 356–367. DOI:<https://doi.org/10.1016/j.chb.2014.07.018>.
- [48] Walonoski, J.A. and Heffernan, N.T. 2006. Prevention of Off-Task Gaming Behavior in Intelligent Tutoring Systems. *International Conference on Intelligent Tutoring Systems* (2006), 722–724.
- [49] Weston, C. et al. 2001. Analyzing Interview Data: The Development and Evolution of a Coding System. *Qualitative Sociology*. 24, 3 (2001), 381–400. DOI:<https://doi.org/10.1023/A:1010690908200>.
- [50] Wijffels, J. et al. 2017. *Package ‘udpipe’*.
- [51] Winne, P.H. 2010. Improving Measurements of Self-Regulated Learning. *Educational Psychologist*. 45, 4 (Oct. 2010), 267–276. DOI:<https://doi.org/10.1080/00461520.2010.517150>.
- [52] Winne, P.H. 2005. Key Issues in Modeling and Applying Research on Self-Regulated Learning. *Applied Psychology*. 54, 2 (2005), 232–238. DOI:<https://doi.org/10.1111/j.1464-0597.2005.00206.x>.
- [53] Winne, P.H. 2017. Learning Analytics for Self-Regulated Learning. *Handbook of Learning Analytics*. (2017), 531–566. DOI:<https://doi.org/10.18608/hla17.021>.
- [54] Winne, P.H. et al. 2019. nStudy: Software for Learning Analytics about Processes for Self-Regulated Learning. *Journal of Learning Analytics*. 6, 2 (Jul. 2019), 95–106. DOI:<https://doi.org/10.18608/jla.2019.62.7>.

- [55] Winne, P.H. and Hadwin, A.F. 1998. Studying as Self-Regulated Learning. *Metacognition in Educational Theory and Practice*. (1998), 277–304.
- [56] Zimmerman, B.J. 2000. Attaining self-regulation: A social cognitive perspective. *Handbook of Self-Regulation*. (2000), 13–39. DOI:<https://doi.org/10.1016/B978-012109890-2/50031-7>.
- [57] Zimmerman, B.J. 1990. Self-Regulated Learning and Academic Achievement: An Overview. *Educational Psychologist*. 25, 1 (Jan. 1990), 3–17. DOI:https://doi.org/10.1207/s15326985ep2501_2.

SQL-DP:A Novel Difficulty Prediction Framework for SQL Programming Problems

Jia Xu^{*}
Guangxi University
Guangxi, China
xujia@gxu.edu.cn

Tingting Wei
Guangxi University
Guangxi, China
tingtingwei@st.gxu.edu.cn

Pin Lv[†]
Guangxi University
Guangxi, China
lvpin@gxu.edu.cn

ABSTRACT

In an Intelligent Tutoring System (ITS), problem (or question) difficulty is one of the most critical parameters, directly impacting problem design, test paper organization, result analysis, and even the fairness guarantee. However, it is very difficult to evaluate the problem difficulty by organized pre-tests or by expertise, because these solutions are labor-intensive, time-consuming, leakage-prone, or subjective in some way. Thus, it is of importance to automatically evaluate problem difficulty via information technology. To this end, we propose a novel difficulty prediction framework, named SQL-DP, for Structured Query Language (SQL) programming problems, mastering which plays a vital role in learning the database technology. In SQL-DP, semantic features of problem stems and structure features of problem answers in the form of SQL codes are both computed at first, using the NLP and the neural network techniques. Then, these features are used as the input to train a difficulty prediction model with the statistic error rates in tests as the training labels, where the whole modeling does not introduce any experts, some as knowledge labeling. Finally, with the trained model, we can automatically predict the difficulty of each SQL programming problem. Moreover, SQL programming problem answering log data of hundreds of undergraduates from Guangxi University of China are collected, and the experiments conducted on the collected log data demonstrate the proposed SQL-DP framework outperforms the state-of-the-art solutions apparently. In particular, SQL-DP decreases the RMSE of difficulty prediction by at most 7.23%, compared with the best-related framework.

Keywords

Problem difficulty prediction, SQL programming problem, Code structure, Neural network, Intelligent tutoring system

1. INTRODUCTION

^{*}Corresponding author.

[†]Corresponding author.

J. Xu, T. Wei, and P. Lv. SQL-DP: A novel difficulty prediction framework for SQL programming problems. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 86–97, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852986>

Stem:	Query the name of students who have taken the course with the course id 'C1'.
Answer:	SELECT Sname FROM Student WHERE Sno IN (SELECT Sno FROM SC WHERE Cno = 'C1');
Difficulty:	0.4

Figure 1: An example of SQL programming problem.

With the progress of information technology, Intelligent Tutoring System (ITS) services are broadly applied, where problem (or question) difficulty has become one of the most critical parameters. The problem difficulty refers to the percentage of students who wrongly answer the problem [14]. Given the information of problem difficulty, an ITS can recommend exercises of suitable difficulty to students with varied knowledge proficiency [31], can automatically organize a test paper by choosing questions with different difficulty levels [12], and can better achieve a fairness guarantee for various other types of education tasks [29]. However, the difficulty of a problem is not directly observable before the test is conducted. To predict the problem difficulty in advance, traditional methods often resort to expertise (e.g., experienced teachers) who are asked to manually label the question difficulty according to their experience, or artificial tests organization [15]. Unfortunately, these human-based solutions are limited in that they are labor-intensive, time-consuming, leakage-prone, or subjective in some way [16]. Therefore, there is an urgent need to design problem difficulty prediction methods without manual intervention.

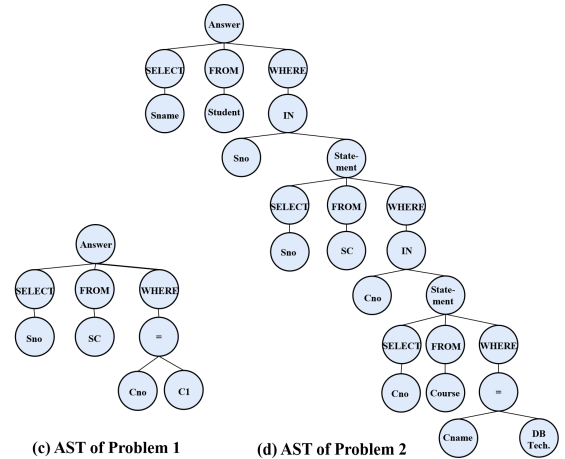
Recently, several non-human-based solutions that rely on machine learning techniques have been proposed [16, 23, 29, 18, 21, 8, 11]. For example, in [16], Huang et al. present TACNN, a test-aware attention-based convolutional neural network to automatically solve the difficulty prediction task for reading comprehension problems in standard English tests. In specific, TACNN utilizes the information of reading passage, question, option, and answer together to predict the difficulty. As another example, Qiu et al. in [23] propose a document enhanced attention-based neural Network (DAN) to predict the difficulty of multiple-choice problems in medical exams. Besides considering stem, option, and answer, DAN fetches relevant medical documents to enrich the information of each question. Moreover, in [29], Tong et al. design a group of salable data-driven models, i.e., C-MIDP and R-MIDP, based on CNN and RNN neural network architectures to predict the difficulty of mathematical

Student					Course			SC		
Sno	Sname	Sex	Age	Dept	Cno	Cname	Credit	Sno	Cno	Grade
S1	Emma	F	20	CS	C1	Database Technology	4	S1	C1	92
S2	Edith	M	19	MA	C2	Information System	4	S2	C2	85

(a) An Instance of Student-Course Database

SQL Programming Problem 1 Stem: Query the id of students who have taken the course with the course id 'C1'. Answer: SELECT Sno FROM SC WHERE Cno = 'C1'; Problem Difficulty: 0.2
SQL Programming Problem 2 Stem: Query the name of students who have taken the course with the course name 'Database Technology'. Answer: SELECT Sname FROM Student WHERE Sno IN (SELECT Sno FROM SC WHERE Cno IN (SELECT Cno FROM Course WHERE Cname = 'Database Technology')); Problem Difficulty: 0.6

(b) Two Example SQL Programming Problems



(c) AST of Problem 1

(d) AST of Problem 2

Figure 2: Examples of SQL programming problems and different code structures of their answers.

questions, which are leveraged by the historical test logs and the corresponding item materials (e.g., stem, option). However, no work is still proposed to estimate the difficulty of Structured Query Language (SQL) programming problems. Structured Query Language (SQL) is the de-facto database query language widely used in industry and taught in almost all computer-related majors in universities [19, 28], we thus focus on solving the difficulty prediction for SQL programming problems in this paper. Figure 1 shows a SQL programming problem containing a stem, an answer, and a difficulty label.

Being different from the existing non-human based solutions which only emphasizes the analysis of stem and option of problems, the difficulty of a SQL programming problem is not only influenced by the stem, options, or answer of the problem, but also depends to a great extent upon the structure of the SQL code answer to the problem. Taking the two SQL programming problems defined over the ‘Student-Course Database’ in Figure 2 as an example. Note that the ‘Student-Course Database’ has three tables, namely Student, Course, and SC, which records student-course selection relationships. Though the two examples of SQL programming problems in the figure have very similar stems, their standard answers in the form of SQL codes exhibit significantly different code structures depicted by Abstract Syntax Trees (ASTs) [17], which makes their difficulty values apparently different from each other.

To this end, we propose SQL-DP, a novel difficulty prediction framework for SQL programming problems and name it as SQL-DP. SQL-DP consists of two major modules. First, the feature extraction module is responsible for computing representations of SQL programming problems by not only considering the semantic information in their stems, but also taking the structure information of their answers in the form of SQL code into account. Using the computed representations of SQL programming problems as the input, then the difficulty prediction module in SQL-DP trains a difficulty prediction model with the statistic error rates in tests as the training labels. The trained difficulty prediction model can be applied to estimate the difficulty of a SQL program-

ming problem in an automatic manner. Note that the whole modeling in SQL-DP does not introduce any human intervention or knowledge labeling, ensuring the usability of the solution in a wide range of application scenarios. The Main contributions of this paper include:

- We propose SQL-DP, a novel difficulty prediction framework for SQL programming problems, which not only considers the semantics of problem stems, but also leverages the information of code structures of problem answers to quantify the difficulty of the problem. As far as we know, this is the first systematic solution to predict the difficulty of SQL programming problems.
- We collect the stems, answers, and answering results of hundreds of SQL programming problems by organizing undergraduates to complete tests of SQL programming problems using our self-developed SQL online judge (OJ) system.
- We conduct a group of experiments using the collected physical-world dataset, and the experimental results show the superiority of the proposed SQL-DP framework in predicting the difficulty values of SQL programming problems.

The rest of our paper is organized as follows. Section 2 discusses the related works; Section 3 states the preliminaries of this work; Section 4 describes details of the proposed SQL-DP framework for the prediction of SQL programming problems; then Section 5 evaluates the effectiveness of the proposed SQL-DP framework using physical-world dataset; finally Section 6 concludes the paper and points out future research directions.

2. RELATED WORKS

Problem (or question) difficulty prediction is an important problem having been widely studied in educational domain.

Traditionally, the difficulty of a question is predicted and labeled by expertise (e.g., experienced teachers) according

to their experience[1], which is labor-intensive and subjective, and not applicable when there are too many questions. An alternative solution is to organize an artificial test (also called as pretest) [1, 15], where a small group of students are required to take part in the artificial test and their error rates are then used to estimate the question difficulty. To improve the estimation precision, a group of educational measurement theories are applied, including *Classical Test Theory* (CTT) [6], *Item Response Theory* (IRT) [9], and *Cognitive Diagnosis Theory* [30]. Many well-known tests, such as Test of English as Foreign Language (TOEFL) and Scholastic Assessment Test (SAT) predict question difficulty following such solution. Unfortunately, the artificial tests based difficulty prediction solution is limited in that it is not only labour-intensive, time-consuming, but also has the risk of question leakage [16].

To overcome the shortcomings of traditional methods, with the development of machine learning technologies, many non-human based solutions of problem difficulty prediction emerge, which can be divided into two categories, i.e., simple regression analysis based and artificial neural network based.

Simple regression analysis based methods establish a simple regression model (e.g., linear regression, multiple regression, and SVM) to construct a mapping function between question difficulty and its influence factors, and the difficulty of new questions can be predicted based on the regression model. As an early example, Chon and Shin [3] built a difficulty prediction model for multiple-choice reading test items by using the multiple regression technique and maximum likelihood estimation. Several features of items, such as response time of testees and paragraph length, are set as the influence factors of item difficulty. A difficulty estimation model based on correlation and regression analysis for English vocabulary questions was then discussed in [27]. As another example, in [25], Makoto Sano proposed to extract a series of language features from multiple-choice questions of reading comprehension, and analyze the extracted features using multiple regression models to obtain the features most related to the question difficulty. Moreover, in [8], Masri et al. proposed to analyze influence factors (e.g., topic and depth of knowledge) of difficulty questions in the sixth grade science test of British primary schools, and establishes a stepwise regression model to predict the difficulty of questions. A difficulty prediction model was also presented towards suggestive blank filling questions for English Tenses [21], which employs the ridge regression model to analyze many factors (e.g., questions text and blank filling words) affecting the question difficulty. These regression analysis based methods, however, have limitations, since they require some domain knowledge and artificially define the factors affecting the question difficulty.

In view of the limitations of simple regression analysis, many researchers proposed to learn the complex relations between influencing factors and question difficulty via artificial neural networks, and as a result achieved the goal of automatic question difficulty prediction. For example, in [16], Huang et al. proposed a model, named TACNN, to predict the difficulty of reading comprehension questions in English test via Convolutional Neural Network (CNN), by using the in-

formation of reading passage, question, options, and answer of each question. Similarly, Tong et al. in [29] proposed a prediction method for the difficulty of mathematical questions based on both of CNN and Recurrent Neural Network (RNN), by analyzing stem, options, and answers of every question. Besides, in [11], Hsu et al. introduced a novel method for automated estimation of multiple-choice items which consist of the following item elements: a question and alternative options. The proposed method utilizes neural network to learn embeddings of question materials in semantic space. Then, it computes the semantic similarities among the stem, answer, and distractors, which are then together fed to a SVM for training the question difficulty prediction model. Then, in [18], Lin et al. proposed a question difficulty prediction model for Chinese reading comprehension problems based on long-term and short-term memory artificial neural network, while in [23] proposed a difficulty prediction method, named as DAN, for multiple-choice questions in medical examination based on neural network model. In specific, besides considering stem, options, and answer, DAN fetches relevant medical documents to enrich the information of each question.

Although recent years have witnessed many works that predict problem difficulty automatically without manual intervention. However, none of these works focus on solving the difficulty prediction of SQL programming problems whose answers in the form of SQL codes are significantly different from their answers. Considering the structure information of SQL codes has great impact on the difficulty of SQL programming problems, all existing works hence can not well handle the difficulty prediction problem for SQL programming items which is discussed in this paper.

3. PRELIMINARY

In this section, we first introduce the method of capturing code structure information. Then we formally define the problem of difficulty prediction for SQL programming problems.

3.1 Code Structure Extraction

In [13], Hindle et al. demonstrate that programming languages, similar to natural languages, also contain abundant statistical properties. However, there are also obvious differences that the code of programming language contains rich and clear structural information between programming language and natural language[20]. By extracting the structure information in the code, we can better analyze the source program. Therefore, some works have studied how to capture the code structure information[2, 20, 22].

In [20], Mou et al. parse code into AST and design a novel Tree-Based Convolutional Neural Network (named as TBCNN) to capture code structural information. In TBCNN model, an AST node is first represented as a distributed, real-valued vector so that the (anonymous) features of the symbols are captured. The vector representations are learned by a coding criterion in [22]. Then Mou et al. design a set of subtree feature detectors, called the tree-based convolution kernel, sliding over the entire AST to extract structural information of a program. Thereafter they apply dynamic pooling [26] to obtain information over different parts of the tree. Finally, a hidden layer and an output layer are added.

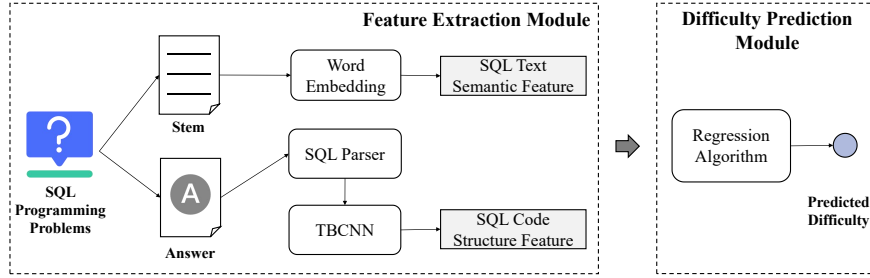


Figure 3: SQL-DP: A difficulty prediction framework for SQL programming problems.

Because the TBCNN model can capture code semantics efficiently, it has become one of the most classical models of code structure feature extraction.

As a programming language, SQL can generate AST through SQL syntax parsers, and then we can use the TBCNN model to extract the structure information of AST, that is, the structure information of SQL code. In short, to obtain more problem information to predict the problem difficulty, except the stem of SQL programming problems, we use the TBCNN model to capture the code structure information from the answers to SQL programming problems.

3.2 Problem Definition

This paper focuses on the difficulty prediction of SQL programming problems. The example of SQL programming problem is shown in Figure 2. The goal of this paper is to train problem difficulty prediction model by using problem stems, answers and real difficulty of SQL programming problems, and then predict the difficulty of new SQL programming problems.

Formally, given the SQL programming problem set $P = \{p_1, p_2, \dots, p_m\}$ and the corresponding real problem difficulty set $D = \{d_1, d_2, \dots, d_m\}$, the goal is to use the above data P and D to train a model \mathcal{M} , and the trained model \mathcal{M} can estimate the difficulty of new SQL programming problems without test logs. Where P includes the problem stem text set $T = \{t_1, t_2, \dots, t_m\}$ and the problem answer set $A = \{a_1, a_2, \dots, a_m\}$. In addition, $p_i = \{t_i, a_i, d_i\}$, $i \in \{1, 2, \dots, m\}$, p_i represents the SQL programming problem i , t_i represents the problem stem text of p_i , a_i represents the answer to SQL programming problem i , d_i represents the real difficulty of p_i , and m represents the total number of SQL programming problems.

In addition, We obtain the real difficulty of each SQL programming problem from the test logs, followed the previous works[23, 16, 29]. Specifically, we calculate the proportion of incorrect answers by dividing the number of students who have answered the problem incorrectly by the number of students who have responded to the problem[23]. The calculation equation of the difficulty of the problem p_i is as follows:

$$d_i = \frac{s_i}{S_i} \quad (1)$$

where d_i is real difficulty of p_i , s_i represents the number of students who have answered p_i incorrectly, and S_i represents

the total number of students who have responded to the problem p_i .

4. SQL-DP FRAMEWORK

This section will describe the difficulty prediction framework of SQL programming problems in detail. As shown in Figure 3, the difficulty prediction framework of SQL programming problems can be divided into two modules: 1) Feature extraction module, which mainly extracts features from the stem and answer of SQL programming problems. The extracted features include stem text semantic features and code structure features; 2) The difficulty prediction module, which uses machine learning to predict the difficulty value of SQL programming problems. Briefly, given the SQL programming problem include stem, answer, and difficulty, we use the feature extraction model to obtain the text semantic features and code structure features from the stems and answers. Then we take the above features and real difficulty as the input of the difficulty prediction module, where the real difficulty is the trained label. Finally, we can use the trained model, the stem and answer of new SQL programming problems to predict the difficulty of new problems, that is, the new SQL programming question without test logs.

4.1 Feature Extraction Module

The feature extraction module consists of SQL text semantic feature extraction module and SQL code structure feature extraction module. The former module uses word embedding techniques to obtain the text semantic features from the stem of SQL programming problems, and the last module extracts SQL code structure features from the answer to problems. The two modules mentioned above will be described in detail below.

4.1.1 SQL Text Semantic Feature Extraction Module

In the task of question difficulty prediction, word embedding technique is often used to obtain the text features of the question, including word2vec, Term Frequency–Inverse Document Frequency (TF-IDF), etc[16, 18, 29]. To extract textual semantic features in SQL programming problems effectively, we adopt various word embedding techniques used in [16] and [29] to extract textual information, including Bag-of-Words (BoW), TF-IDF, and word2vec. Finally, we select the optimal word embedding technique for follow-up experiments.

BoW does not consider the order of words in the sentence.

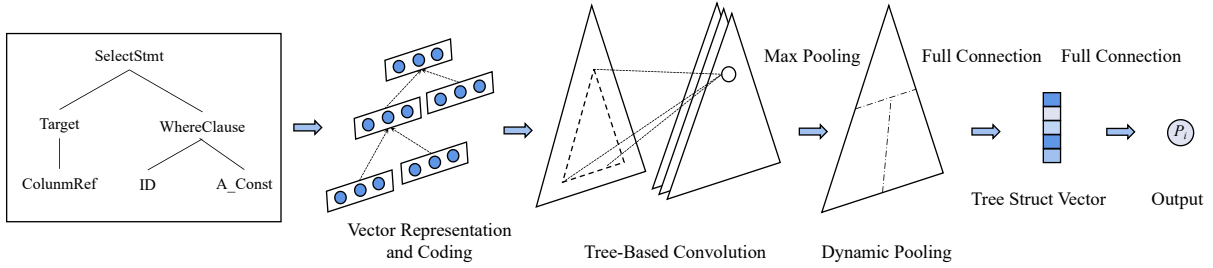


Figure 4: The structure of TBCNN model.

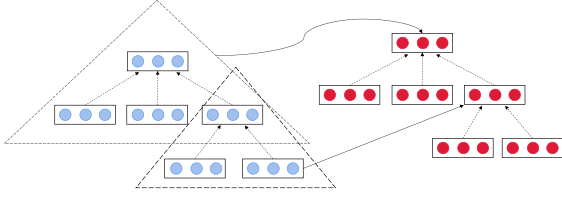


Figure 5: Tree-based convolution.

Still, it only counts the number of occurrences of words, so the value of each position of the text vector calculated by the model is the number of occurrences of the corresponding word. Because the model only records the first occurrence of each word and the number of occurrences of each word, the text features calculated by the model do not contain important information such as the grammar and semantics of the text.

TF-IDF is a word embedding technique used to calculate the importance of each word. The importance of the evaluation word is based on the term frequency and the inverse document frequency of the word appearing in the text. The term frequency counts the number of times each word appears in the text. And the inverse document frequency is used To measure the commonness of each word. The more common words have smaller IDF values.

word2vec is a technique that can efficiently learn the vector representation of text, which can accurately capture the syntactic and semantic word relationships in the text. When using word2vec to extract semantic information in the stem of SQL programming problems, we firstly take all the stem texts of SQL programming problems to train the word2vec model and obtain the word embedding vector of each word or character. Then, we segment the stem, remove the stop word, and replace the remaining words or characters with the corresponding word embedding vector in order to obtain the representation vector of the stem. Next, the input of the difficulty prediction module is a fixed length feature, but the length of words in the different stems is not the same. According to the method of [29], we set the text semantic feature of the stem to a fixed length J . If the length of stem vector is less than J , it is filled with zero. Otherwise, the

word behind J is deleted. Finally, we get the text semantic feature vector of SQL programming problem, which can retain certain semantic features.

4.1.2 SQL Code Structure Feature Extraction Module

Inspired by [20] and [32], in order to capture the tree structure information of the code as much as possible and enrich the input of the difficulty prediction of SQL programming problems, we apply TBCNN model to capture SQL code structural information. The TBCNN model structure is shown in Figure 4.

TBCNN model includes embedding layer, tree-based convolution layer, dynamic pooling layer, and output layer. TBCNN first converts the abstract syntax tree of SQL code into a vector representation suitable for later calculation. Then, local features are extracted by tree-based convolution, and a new set of feature vectors with the same structure as the input tree are obtained. However, the tree structure of different SQL codes will be different. Therefore, in order to input it into the final full connection layer, the tree structure obtained from the convolution layer is simplified into a fixed vector shape through the dynamic pooling layer. Finally, classification or regression is carried out through the full connection layer. This paper applies TBCNN model to the regression problem, and the training label of the model is the real difficulty of the problem.

Specifically, the input of TBCNN model is the serialized abstract syntax tree (AST). Therefore, the model first uses the pre-training method in [22] to obtain the representation vector of nodes in ASTs. The more specific representation method of node vector can be found in [22].

Then, in the tree-based convolution layer, a set of fixed-depth feature detectors that can slide on the whole AST is designed to extract the structure information of the program. The subtree feature detectors can be viewed as convolution with a set of finite support kernels, so the subtree feature detector is called tree-based convolution. The output of the feature detector is calculated by Equation 2.

$$y = \tanh\left(\sum_{i=1}^n W_{conv,i} \cdot x_i + b_{conv}\right) \quad (2)$$

where $y \in R^{N_c}$, x_1, x_2, \dots, x_n is the vector representation

of nodes in the sliding window, and $W_{conv,i} \in R^{N_c \times N_f}$ is the parameter, $b_{conv} \in R^{N_c}$ is the bias, N_c is the number of feature detectors.

In addition, continuous binary tree (as shown in Figure 5) is proposed to handle the different number of child nodes. The convolution layer uses three weight matrices as parameters, including W_{conv}^t , W_{conv}^l , and W_{conv}^r , in which superscript t , l , and r refer to “top”, “left”, and “right” respectively. The weight matrix of node x_i in the sliding window is a linear combination of W_{conv}^t , W_{conv}^l , and W_{conv}^r (as shown in Equation 3). η_i^t , η_i^l , and η_i^r are the coefficients. In this paper, we use the method in [32] to calculate the coefficient η_i^t , η_i^l , and η_i^r , as shown below:

$$W_{conv,i} = \eta_i^t W_{conv}^t + \eta_i^l W_{conv}^l + \eta_i^r W_{conv}^r \quad (3)$$

$$\eta_i^t = \frac{d_{max} - d_i}{d_{max}} \quad (4)$$

$$\eta_i^l = \begin{cases} \frac{i-1}{n-1}(1 - \eta_i^t) & \text{non-leaf} \\ 0.5(1 - \eta_i^t) & \text{leaf} \end{cases} \quad (5)$$

$$\eta_i^r = (1 - \eta_i^t)(1 - \eta_i^l) \quad (6)$$

where d_{max} represents the depth of the sliding window and d_i represents the depth of node i in the sliding window.

After convolution, structural features in an AST are extracted, but the features are still a tree structured set of vectors, which can not be directly fed into the fully connected layer. Therefore, dynamic pooling [26] is applied to reduce the tree into a fixed shape vector.

Finally, a full connection layer and an output layer are added to predict the difficulty of SQL programming problems. The difficulty prediction task of SQL programming problems in this paper belongs to regression problem, inspired by [29], we construct the loss function of the TBCNN model as the Equation 7.

$$L = \frac{1}{n} \sum_{i=1}^n (\tilde{p}_i - d_i)^2 \quad (7)$$

where \tilde{p}_i and d_i represent the predicted difficulty value and the real difficulty value of the problem p_i respectively.

4.2 Difficulty Prediction Module

The difficulty of SQL programming problems can be predicted using various regression models. Therefore, the difficulty prediction module of SQL programming problems uses a variety of regression models to predict the difficulty of SQL programming problems to obtain the optimal regression model. Regression models used include linear regression (LR) [4], support vector machine (SVM) [7], gradient boosting decision tree (GBDT) [10], random forest (RF) [5] and back propagation (BP) [24] neural networks. The linear regression model is one of the basic regression models commonly used in linear regression problems, but it cannot solve nonlinear distribution problems. The SVM model is common in both linear and nonlinear problems. The RF model is a kind of integrated algorithm with the advantages of difficult overfitting, strong anti-noise ability, and strong interpretability. The GBDT model is an iterative decision

Table 1: The statistics of the dataset.

Statistics	Value
# of answer logs	10952
# of students	283
# of SQL programming problems	318
Average answer logs per question	34.4
Average answer logs per student	38.7

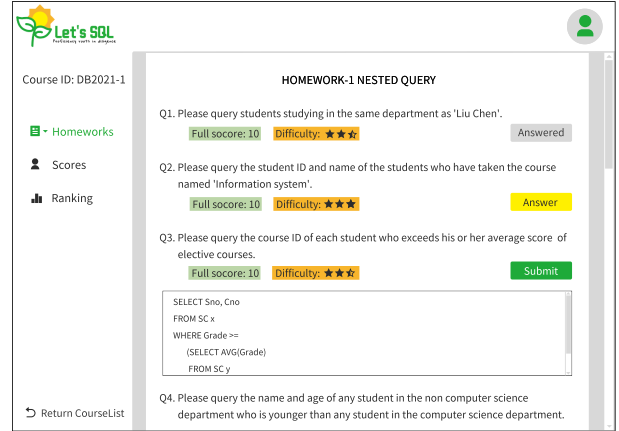


Figure 6: A graph for self-developed SQL online judge system.

tree algorithm that can be used for regression tasks. The BP neural network refers to the multilayer feedforward neural network trained by the BP algorithm. It has strong representation ability and nonlinear mapping ability, but its strong learning representation ability makes it easy to overfit.

The following describes the specific process of the difficulty prediction module:

- Firstly, the text semantic features and code structure features of the extracted SQL programming problems are concatenated together and used as the input of the above regression model. The regression models are trained using the actual difficulty of the SQL programming problems as the training labels.
- Secondly, the model with the best results among the regression models mentioned above is selected as the regression model of the difficulty prediction module of SQL programming problems.
- Thirdly, input the text features and code structure features of the SQL programming problems that need to predict difficulty into the trained regression model. The difficulty of the SQL programming problems can be obtained.

5. EXPERIMENTS

In this section, we first introduce the source of the dataset. Then we raise the evaluation metrics and experimental comparison methods used in the experiment. Next, we present the experimental settings in detail. Finally, we summarize and analyze the experimental results.

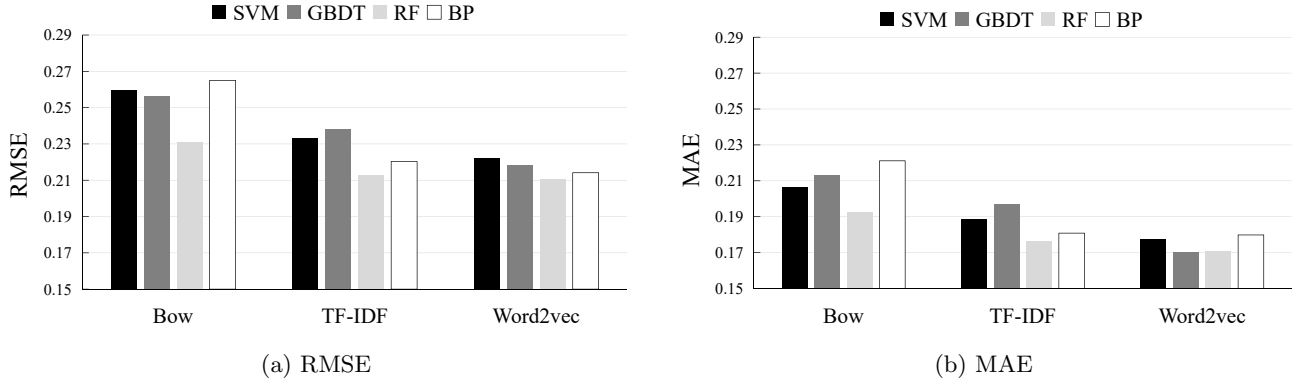


Figure 7: Results of text semantic features obtained by different word embedding technologies on different algorithms.

5.1 Dataset Description

The dataset in this paper comes from our self-developed SQL OJ system (as shown in Figure 6). The data collection lasted for two years, involving 306 students from Guangxi University of China. In addition, if only a small count of students have tried to solve a problem, the obtained difficulty of this problem will have severe randomness[23]. Therefore we use the processing method in [23] to process our data. Specifically, we eliminate the problems having no more than ten test logs, and the detail of dataset after processing is shown in Table 1.

5.2 Evaluation Metrics

Followed the previous works[23, 29, 16], we use the evaluation metrics commonly used in question difficulty prediction: *Root Mean Square Error* (RMSE) and *Mean Absolute Error* (MAE). The above two evaluation metrics are widely used in question difficulty prediction to measure the distance between the predicted difficulty value and the actual difficulty value. The smaller the evaluation metric value is, the better performance the results have. The two evaluation metrics are shown in Equation 8 and Equation 9 respectively:

$$\text{RMSE} = \sqrt{\frac{1}{M} \sum_{i=1}^M (\tilde{p}_i - d_i)^2} \quad (8)$$

$$\text{MAE} = \frac{1}{M} \sum_{i=1}^M |\tilde{p}_i - d_i| \quad (9)$$

where M represents the number of SQL programming problems, and \tilde{p}_i and d_i represent the predicted difficulty and real difficulty of programming problem i respectively.

5.3 Comparison methods

The difficulty prediction of questions mentioned in the previous related work is not for SQL programming problems. Therefore, we modify C-MIDP model, R-MIDP model, and H-MIDP model of the mathematical questions proposed in [29] to adapt to the SQL programming problems. The above three models first use the word2vec to vectorize the texts of mathematical questions and use the scoring rate of the students in the mathematical questions as the actual difficulty value of the mathematical questions. Then take the

real difficulty value of the mathematical questions as the training labels. Finally, input the text representation vector of the mathematical problem into different neural networks for learning. The main processes of the C-MIDP, R-MIDP and H-MIDP models (The structure diagram of C-MIDP, R-MIDP, and H-MIDP is presented in the appendix Table 10) are given below .

- First, use the word2vec model to train all the texts of the mathematical questions, and then the text representation vector of the math questions can be obtained.
- Secondly, the text representation vector of the mathematical question is used as the models' input. The actual difficulty of the question is the scoring rate of each question, and it is used as the training label of the model. During the training process, different models extract different mathematical question information.
- Thirdly, considering that the data in the dataset comes from many different schools. And the student groups in other schools have differences in the knowledge level status, which will cause the scoring rate of the question (i.e., the actual difficulty value) to be affected by the difference in the level of the student group. To eliminate this effect, a context-related training method is proposed. That is, a new loss function is constructed for training the model.
- Finally, for new math questions for which there is no student answer data or insufficient student answer data, the text vector of the new question can be input into the trained model to predict the difficulty of the question.

The C-MIDP, R-MIDP, and H-MIDP models have the same process, and the differences between the three models are: The C-MIDP model uses a multi-layer Convolutional Neural Network to mine different levels of text semantic information from the text of mathematical questions. The R-MIDP model uses the recurrent neural network (RNN) suitable for mining long-range logical relations to mine the sequential logical information of mathematical problems from the text. The H-MIDP model combines the advantages of the

Table 2: TBCNN’s hyperparameters.

Hyperparameter	Value
Initial learning rate	0.001
Learning rate decay	0.001
Node embedding dimension	100
Convolutional layers’ dimension	50

C-MIDP and the R-MIDP and can simultaneously extract crucial semantic information and sequential logic information of mathematical problem text.

5.4 Experimental Setup

Word Embedding: In [16] and [29], Huang et al. and Tong et al. use BoW, TF-IDF, and word2vec technologies to obtain the text semantic features of questions. Following their works, we use the aforementioned methods to process the text semantic features. The Bow and TF-IDF methods are relatively simple, so we only introduce the setting of the word2vec in detail. Specifically, the corpus used for word2vec training is all the stem texts in the dataset. The maximum number of words after word segmentation in the problem stem is 17, so we set the number of words in each problem to 17, and the word vector dimension of each word is 50. Therefore, the stem text vector of each problem has a dimension of 850.

TBCNN Setting: The input of TBCNN model is the AST of programming code. So we first use SQL parser `pglast`¹ to parse the SQL codes, and then we obtain the AST of the codes. After that, we serialize the ASTs and take the serialized ASTs as the input of the TBCNN model. Besides, TBCNN’s hyperparameters are shown in Table 2.

Other Setting: All models in the experiment use ten-fold cross-validation to verify the performance of the model. In experiment, the programming language we used is python, and the experiment is configured with 2-core CPU, 8GB memory, 1TB hard disk, and 64-bit Ubuntu operating system.

5.5 Experimental Results

In this section, we run an ablation study to highlight the individual contribution of each module in SQL-DP and compare the SQL-DP proposed in this paper with the C-MIDP, R-MIDP, and H-MIDP model proposed in [29].

5.5.1 Text Semantic Feature Extraction

A variety of word embedding techniques can be used in the SQL-DP framework proposed in this paper, so we need to experiment with different word embedding techniques to obtain the optimal problem text semantic features for subsequent experiments.

Figure 7 shows the results of text semantic features obtained using different word embedding techniques on various machine learning algorithms. But the RMSE of the LR algorithm exceeds 0.5, and the result on the MAE is also poor, so we do not show the results of LR algorithm in Figure 7 in

¹<https://pglast.readthedocs.io/en/v3/index.html>

Table 3: Experimental results of regression model selection

Models	RMSE	MAE
SVM	0.2176	0.1744
GBDT	0.2128	0.1721
RF	0.1977	0.1570
BPNN	0.2022	0.1702

order to a more intuitive display. And the poor experimental results of LR also prove that LR is not competent for difficulty prediction problem. Therefore, we will no longer use the LR algorithm to predict the difficulty of SQL programming problems in the subsequent experiments.

In addition, as shown in Figure 7, the text semantic features extracted by BoW perform the worst in predicting the difficulty of SQL programming problems, while the overall performance of word2vec is the best. This shows that the word2vec is more suitable for extracting text semantic features of problem stems than BoW and TF-IDF for SQL programming problems. Therefore, we will use word2vec to obtain the text semantic features of SQL programming problems in the subsequent experiments.

5.5.2 Regression Model Selection Experiment

SQL-DP can use a variety of regression models to predict the difficulty of SQL programming problems. Therefore, we experiment with multiple regression models to select the best model for subsequent experiments. In the regression model selection experiment, text semantic features and code structure features are used as the input of the regression model at the same time. Table 3 shows the results of using the two features mentioned above as the input of multiple regression models simultaneously. As can be seen from Table 3, when both text semantic features and code structure features are used as the input of the regression model, the results of SQL-DP using the SVM model are the worst, while the effects of the RF model are the best. Therefore, in the subsequent comparative experiment, the SQL-DP framework will use the RF model to predict the difficulty of SQL programming problems.

5.5.3 Comparative Experiment

In this section, we compare SQL-DP with C-MIDP, R-MIDP, and H-MIDP to prove the effectiveness of the difficulty prediction framework of SQL programming problems proposed in this paper. Given the above text semantic feature extraction experiment and ablation experiment results, we choose the SQL-DP framework using word2vec, TBCNN, and RF algorithm to compare with the C-MIDP, R-MIDP, and H-MIDP.

Table 4 shows the experimental results of SQL-DP, C-MIDP, R-MIDP, and H-MIDP. We can see from Table 4 that results of SQL-DP with respect to the two evaluation metrics, i.e., RMSE and MAE, are consistently better than those of C-MIDP, R-MIDP, and H-MIDP models, which verifies the superiority of the proposed SQL-DP in predicting the difficulty of SQL programming problems. Moreover, we are delighted to see from the table that SQL-DP increases the RMSE by 7.23%, compared with the best comparison model H-MIDP,

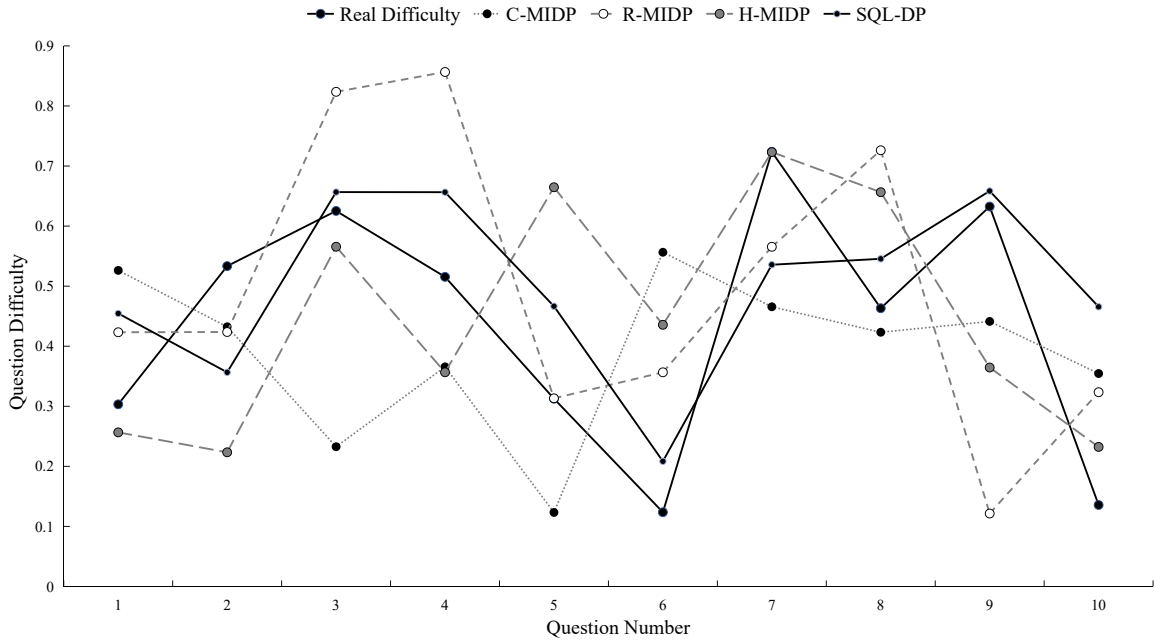


Figure 8: Comparison between the difficulty predicted by four models and the ground truth on ten SQL programming problems.

which is a great progress made under the context of difficulty prediction for SQL programming problems. Because the C-MIDP, R-MIDP, and H-MIDP models only extract information from the text of SQL programming problems and do not extract the code structure information of the answers. Unlike those three comparison models, in SQL-DP, in addition to the text semantic feature extraction module, we can obtain rich semantic information from the stem of SQL programming problems. The code structure extraction module can also obtain rich code structure information from the answers to SQL programming problems. It is the consideration of code structure information in SQL-DP that further enhances the difficulty prediction ability of it towards SQL programming problems, compared with state-of-the-art models.

In addition, we can observe from Table 4 that the H-MIDP model performs best among all state-of-the-art models, while the R-MIDP model performs worst among the three models C-MIDP, R-MIDP, and H-MIDP. The reason may be that the RNN in the R-MIDP model is better at capturing the logical relationship in long sentences. But in the SQL programming problem dataset collected in this paper, the descriptions of the stem of the SQL programming problems are generally short, so the performance of the R-MIDP model on the SQL programming problems is the worst.

To more intuitively see the advantages of our proposed SQL-DP in predicting the difficulty of SQL programming problems, we randomly select 10 problems in the test set. And we test them with the trained C-MIDP model, R-MIDP model, H-MIDP model, and our proposed SQL-DP. After that, we use a broken line diagram (as shown in Figure 8) to show the distance between the predicted problem difficulty of the above models and the real problem difficulty. As shown in

Table 4: Comparative experimental results.

Models	RMSE	MAE
C-MIDP	0.2167	0.1603
R-MIDP	0.2228	0.1785
H-MIDP	0.2131	0.1578
SQL-DP	0.1977	0.1570

Table 5: Ablation experimental results of SQL-DP.

Feature	RMSE	MAE
Text semantic feature	0.2106	0.1711
Code structure feature	0.2124	0.1721
Text semantic + Code structure	0.1977	0.1570

Figure 8, we can see that the prediction difficulty of SQL-DP for the selected SQL programming problems is closer to the real problem difficulty than the C-MIDP, R-MIDP, and H-MIDP models.

5.5.4 Ablation Experimental

To get deep insights into the contributions of various modules in the SQL-DP framework proposed in this paper, we also conduct some ablation prediction outcomes.

As shown in Table 5, we can observe a performance decrease by removing the SQL text semantic feature extraction module or the SQL code structure feature extraction module. Besides, we can also see that the overall performance of the SQL text semantic feature extraction module is similar to that of the SQL code structure feature extraction module. This observation shows that both SQL code structure fea-

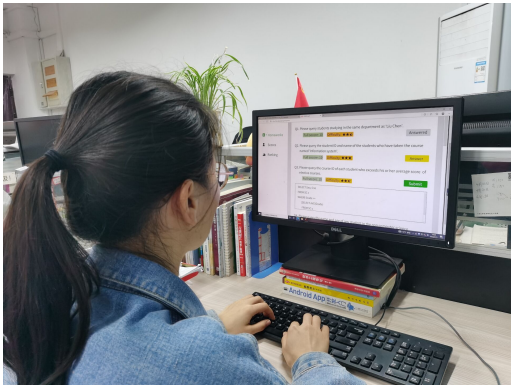


Figure 9: Scenario of a student using self-developed SQL OJ system.

tures and SQL text semantic features play an essential role in the difficulty prediction of SQL programming problems.

5.5.5 Application

Given the effectiveness of the SQL-DP framework proposed in this paper, we use the trained SQL-DP to predict the difficulty of new SQL programming problems. That is, the problem without student test logs. And apply it to our self-developed SQL OJ system, as shown in Figure 6. Specifically, we map the problem difficulty value from 0 to 1 to 5 stars. Among them, the problem difficulty value from 0 to 0.1 is half a star, value from 0.1 to 0.2 is one star, and so on. Figure 9 shows a student practicing SQL programming problems using our self-developed SQL OJ system. The student can choose to do simple SQL programming problems first according to the problem difficulty labels or challenge herself to choose more difficult SQL programming problems in our system.

6. CONCLUSIONS

Conclusions. In this paper, we propose SQL-DP, a novel framework that automatically predicts difficulty of SQL programming problems. SQL-DP makes full use of the information in problem stems and problem answers in the form of SQL codes, based on which the difficulty values of SQL programming problems can be effectively estimated using machine learning techniques. Besides, we organized many tests of SQL programming problems in real teaching practice, which last for two years and involve seven different undergraduate classes in Guangxi University of China, and collected a SQL dataset containing materials and student answering logs of hundreds of SQL programming problems. Experimental results over our collected physical-world SQL dataset show that the proposed SQL-DP gains apparently better prediction performance towards SQL programming problems, compared with state-of-the-art solutions.

Generalization. Though SQL-DP is discussed for the difficulty prediction of SQL programming problems, it can be easily generalized to address the difficulty prediction of programming problems using other languages (such as C and Java), since the consideration of structure information of problem answers is also very important for these programming problems.

Future Works. Two important directions for future works can be considered. First, we will consider more features of SQL programming problems, such as equivalent answers, SQL concepts (e.g., nested queries, multiple tables), etc. Second, to support the difficulty prediction task of programming problems corresponding to more types of programming languages, we will modify and adapt the proposed SQL-DP framework, including the design or use of neural network layers.

Related Resources. To better promote related study of SQL programming problems, the source code of the proposed SQL-DP framework and partial of our collected SQL dataset used in the experiment are all released and can be assessed though the link below: <https://github.com/SQL-DP/SQL-DP>

7. ACKNOWLEDGMENTS

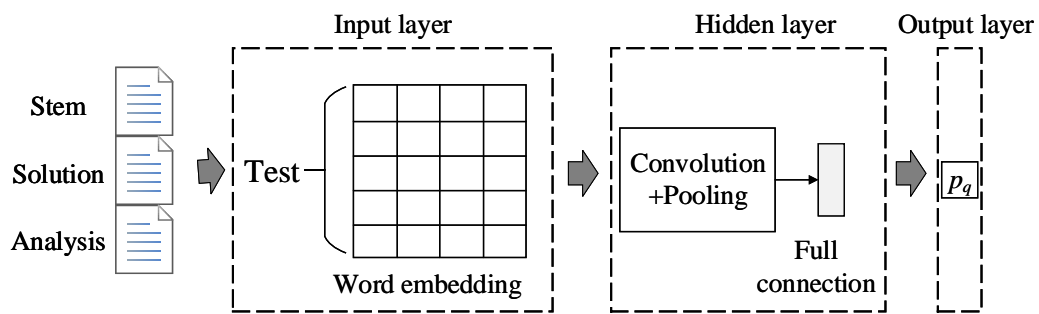
This work is supported by the National Natural Science Foundation of China (No. 62067001), the Projects of Higher Education Undergraduate Teaching Reform Project in Guangxi (Nos. 2017JGZ103 and 2020JGA116), Innovation Project of Guangxi Graduate Education (No. JGY2021003), and the Special funds for Guangxi BaGui Scholars. This work is partially supported by the Guangxi Natural Science Foundation (No. 2019JJA170045). We would like to thank Dr. Wu, Dr. Tian, and Mr. Zhang for providing data from their course teaching practices. Finally, thank Aetf for providing some source code.

8. REFERENCES

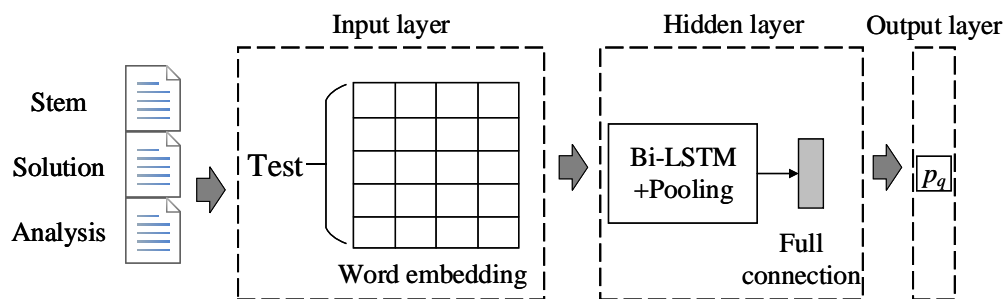
- [1] S. Alkhuzaey, F. Grasso, T. R. Payne, and V. A. M. Tamma. A systematic review of data-driven approaches to item difficulty prediction. In *Proceedings of the 22nd International Conference on Artificial Intelligence in Education*, pages 29–41. Springer, June 2021.
- [2] N. D. Q. Bui, Y. Yu, and L. Jiang. Treecaps: Tree-based capsule networks for source code processing. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, pages 30–38. AAAI Press, February 2021.
- [3] Y. V. Chon and T. Shin. Item difficulty predictors of a multiple-choice reading test. *ENGLISH TEACHING*, 65(4):257–282, December 2010.
- [4] D. R. Cox. Corrigenda: The regression analysis of binary sequences. *Journal of the Royal Statistical Society*, 21(1):238, 1959.
- [5] A. Cutler, D. R. Cutler, and J. R. Stevens. *Random forests*. Springer US, Boston, 2004.
- [6] DeVellis and F. Robert. Classical test theory. *Medical Care*, 44(3):S50–9, December 2006.
- [7] H. Drucker, C. J. C. Burges, L. Kaufman, A. J. Smola, and V. Vapnik. Support vector regression machines. In *Proceedings of the 9th International Conference on Neural Information Processing Systems*, pages 155–161. MIT Press, December 1996.
- [8] Y. H. El Masri, S. Ferrara, P. W. Foltz, and J. A. Baird. Predicting item difficulty of science national curriculum tests: The case of key stage 2 assessments. *The Curriculum Journal*, 28(1):59–82, November 2017.
- [9] X. FAN. Item response theory and classical test theory: An empirical comparison of their item/person

- statistics. *Educational and Psychological Measurement*, 58(3):357–381, June 1998.
- [10] J. H. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29(5):1189–1232, November 2001.
- [11] H. Fu-Yuan, L. Hahn-Ming, C. Tao-Hsing, and S. Yao-Ting. Automated estimation of item difficulty for multiple-choice tests: An application of word embedding techniques. *Information Processing & Management*, 54(6):969–984, January 2018.
- [12] G. Y. Han and X. Z. Li. An intelligent test paper generation algorithm based on adjustment of overall difficulty degrees. *Applied Mechanics and Materials*, 411-414(APR.):2879–2882, September 2013.
- [13] A. Hindle, E. T. Barr, Z. Su, M. Gabel, and P. T. Devanbu. On the naturalness of software. In *34th International Conference on Software Engineering*, pages 837–847. IEEE Computer Society, June 2012.
- [14] P. W. Holland and D. T. Thayer. An alternate definition of the ets delta scale of item difficulty. *ETS Research Report Series*, 1985(2):i–10, December 1985.
- [15] P. Hontangas, V. Ponsoda, J. Olea, and S. L. Wise. The choice of item difficulty in self-adapted testing. *European Journal of Psychological Assessment*, 16(1):3–12, 2000.
- [16] Z. Huang, Q. Liu, E. Chen, H. Zhao, M. Gao, S. Wei, Y. Su, and G. Hu. Question difficulty prediction for READING problems in standard tests. In *Proceedings of the 31th AAAI Conference on Artificial Intelligence*, pages 1352–1359. AAAI Press, February 2017.
- [17] P. Klein, S. Tirthapura, D. Sharvit, and B. Kimia. A tree-edit-distance algorithm for comparing simple, closed shapes. In *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms*, page 696–704. Society for Industrial and Applied Mathematics, January 2000.
- [18] L. Lin, T. Chang, and F. Hsu. Automated prediction of item difficulty in reading comprehension using long short-term memory. In *International Conference on Asian Language Processing*, pages 132–135. IEEE, November 2019.
- [19] A. Lino, A. Rocha, L. Macedo, and A. Sizo. Application of clustering-based decision tree approach in sql query error database. *Future generation computer systems*, 93(APR.):392–406, April 2019.
- [20] L. Mou, G. Li, L. Zhang, T. Wang, and Z. Jin. Convolutional neural networks over tree structures for programming language processing. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, pages 1287–1293. AAAI Press, February 2016.
- [21] I. Pandarova, T. Schmidt, J. Hartig, A. Boubekki, R. D. Jones, and U. Brefeld. Predicting the difficulty of exercise items for dynamic difficulty adaptation in adaptive language tutoring. *Int. J. Artif. Intell. Educ.*, 29(3):342–367, 2019.
- [22] H. Peng, L. Mou, G. Li, Y. Liu, L. Zhang, and Z. Jin. Building program vector representations for deep learning. In *Proceedings of the 8th International Conference on Knowledge Science, Engineering and Management*, pages 547–553. Springer, October 2015.
- [23] Z. Qiu, X. Wu, and W. Fan. Question difficulty prediction for multiple choice problems in medical exams. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 139–148. ACM, November 2019.
- [24] D. E. Rumelhart and J. L. McClelland. *Parallel distributed processing*. MIT Press, Cambridge, 1986.
- [25] M. Sano. Improvements in automated capturing of psycho-linguistic features in reading assessment text. In *The annual meeting of National Council on Measurement in Education*, pages 1–27. National Council on Measurement in Education, April 2016.
- [26] R. Socher, E. H. Huang, J. Pennington, A. Y. Ng, and C. D. Manning. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Proceedings of the 24th International Conference on Neural Information Processing Systems*, pages 801–809. Curran Associates Inc., December 2011.
- [27] Y. Susanti, H. Nishikawa, T. Tokunaga, and O. Hiroyuki. Item difficulty analysis of english vocabulary questions. In *Proceedings of the 8th International Conference on Computer Supported Education*, page 267–274. SciTePress, April 2016.
- [28] T. Taipalus. *Teaching tip: A notation for planning SQL queries*. *J. Inf. Syst. Educ.*, 30(3):160–166, Winter 2019.
- [29] W. Tong, F. Wang, Q. Liu, and E. Chen. Data driven prediction for the difficulty of mathematical items. *Journal of Computer Research and Development*, 56(5):1007–1019, May 2019.
- [30] J. D. L. TORRE. Dina model and parameter estimation: A didactic. *Journal of Educational and Behavioral Statistics*, 34(1):115–130, March 2009.
- [31] Z. Wu, M. Li, Y. Tang, and Q. Liang. Exercise recommendation based on knowledge concept prediction. *Knowledge-Based Systems*, 210:106481, December 2020.
- [32] P. Yu and S. Wang. Deep tree: Sql injection detection by the power of deep learning. <https://github.com/Aetf/tensorflow-tbcnn/blob/master/misc/deeptree.pdf>, 2017.

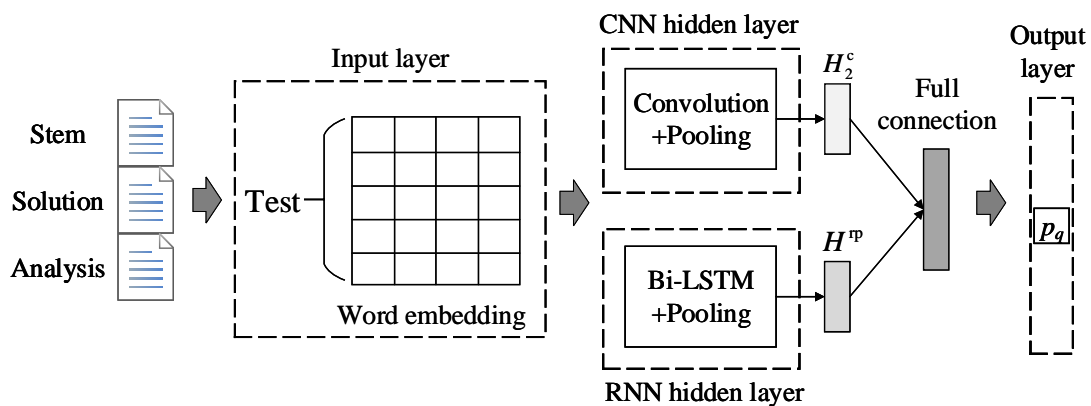
APPENDIX



(a) Structure diagram of C-MIDP model



(b) Structure diagram of R-MIDP model



(c) Structure diagram of H-MIDP model

Figure 10: Structure diagram of C-MIDP, R-MIDP, and H-MIDP model.

Evaluating the Explainers: Black-Box Explainable Machine Learning for Student Success Prediction in MOOCs

Vinitra Swamy
EPFL
vinitra.swamy@epfl.ch

Bahar Radmehr
Sharif University
radmehr.bahar@ee.sharif.edu

Natasa Krco
EPFL
natasa.krco@epfl.ch

Mirko Marras
University of Cagliari
mirko.marras@acm.org

Tanja Käser
EPFL
tanja.kaeser@epfl.ch

ABSTRACT

Neural networks are ubiquitous in applied machine learning for education. Their pervasive success in predictive performance comes alongside a severe weakness, the lack of explainability of their decisions, especially relevant in human-centric fields. We implement five state-of-the-art methodologies for explaining black-box machine learning models (LIME, PermutationSHAP, KernelSHAP, DiCE, CEM) and examine the strengths of each approach on the downstream task of student performance prediction for five massive open online courses. Our experiments demonstrate that the families of explainers **do not agree** with each other on feature importance for the same Bidirectional LSTM models with the same representative set of students. We use Principal Component Analysis, Jensen-Shannon distance, and Spearman’s rank-order correlation to quantitatively cross-examine explanations across methods and courses. Furthermore, we validate explainer performance across curriculum-based prerequisite relationships. Our results come to the concerning conclusion that the choice of explainer is an important decision and is in fact paramount to the interpretation of the predictive results, even more so than the course the model is trained on. Source code and models are released at <http://github.com/epfl-ml4ed/evaluating-explainers>.

Keywords

Explainable AI, LIME, SHAP, DiCE, CEM, Counterfactuals, MOOCs, LSTMs, Student Performance Prediction

1. INTRODUCTION

The steep rise in popularity of neural networks has been closely mirrored by the adoption of deep learning for education. For the majority of educational data modeling tasks such as student success prediction (e.g., [1]), estimating early dropout (e.g., [2]), and knowledge tracing (e.g., [3, 4]), the recent literature relies on neural networks to reduce human involvement in the pipeline and boost overall prediction ac-

curacy. Unfortunately, these advances come at a significant cost: traditional machine learning techniques (e.g., linear regression, SVMs, decision trees) are simple, but interpretable, where deep learning techniques trade transparency for the ability to capture complex data representations [5].

There is a compelling need for interpretability in models dealing with human data, especially in education. [6] emphasizes that explainability and accountability should be incorporated in machine learning system design to meet social, ethical and legislative requirements. Other work [7] strongly argues for the necessity of interpretable models in education, specifically in settings where students can see the effect of a decision but not the reasoning behind it (e.g., Open Learner Models). Predictions of student performance are often used to determine underachieving students for targeted downstream interventions. Identifying important features motivating failure or dropout predictions is crucial in designing effective, personalized interventions.

However, there exists only a handful of papers focusing on explainability in the field of machine learning for education. For example, [8] examined the inner workings of deep learning models for knowledge tracing through layer-relevance propagation. Other researchers [9] experimented with traditional machine learning models for student success prediction and implemented local explanations with LIME for transparency in the best performing model. Additionally, [10] used SHAP feature importances to interpret student dropout prediction models. [11] suggested interventions for wheel-spinning students based on Shapley values. Finally, [12] explored LIME on ensemble machine learning methods for student performance prediction, [13] integrated LIME explanations in student advising dashboards, and [14] used LIME for interpreting models identifying at-risk students.

While field of neural network explainability is also nascent in the broader machine learning community, the last five years have shown a sharp increase in research and industry interest in this topic. Local, instance-based explainability methods like LIME [15] and SHAP [16] have become immensely popular. These methods have been successfully applied on models predicting ICU mortality [17], non-invasive ventilation for ALS patients [18], and credit risk [19]. Recent work in counterfactual explanations [20, 21, 22] searches for a minimal subset of features that leads to the prediction alongside a minimal feature subset that needs to be changed for the

V. Swamy, B. Radmehr, N. Krco, M. Marras, and T. Käser. Evaluating the explainers: Black-box explainable machine learning for student success prediction in MOOCs. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 98–109, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852964>

prediction to change. Counterfactuals have been used in tasks like image classification [23], loan repayment [24], and grouping websites into topics for safe-advertising [25].

Although the explainability corpora is growing, there is a clear gap in explainability literature for education, with an even more pressing need for work (quantitatively) comparing different explainability methods. To the best of our knowledge, current research on explainability in education is exclusively applied: the majority of previous research implements only one specific explainability method to interpret the predictions of their proposed approach.

To address this research gap, we examine and compare five popular instance-based explainability methods on student success prediction models for five different massive open online courses (MOOCs). We formulate comparable feature importance scores for each explainer, scaled between $[0, 1]$ on a uniformly sampled, stratified representative set of students. To quantitatively compare the feature importance distributions, we propose the use of different measures: rank-based metrics (Spearman’s rank-order correlation), distance metrics (Jensen Shannon Distance), and dimensionality analysis (Principal Component Analysis). We validate the explanations through an analysis of feature importance on a MOOC with known prerequisite relationships in the underlying curriculum. With our experiments, we address three research questions: 1) How similar are the explanations of different explainability methods for a specific course (**RQ1**)? 2) How do explanations (quantitatively) compare across courses (**RQ2**)? 3) Do explanations align with prerequisite relations in a course curriculum (**RQ3**)?

Our results demonstrate that the feature importance distributions extracted by different explainability methods for the same model and course differ significantly from each other. When comparing the feature importances across courses, we see that LIME is far apart from all other methods due to selecting a sparse feature set. Furthermore, our findings show that the choice of explainability method influences the feature importance distribution much more than the course the model is predicting on. Our examination on prerequisite relationships between features further indicates that the three families of methods are only partially able to uncover prerequisite dependencies between course weeks. Source code and models are released on Github¹.

2. METHODOLOGY

The goal of this paper is to compare explanations from deep learning models tasked with identifying student success prediction in MOOCs. In this section, we formalize the student success prediction task addressed in this paper including the data collection and preprocessing, feature extraction, and model preparation. We then introduce the considered explainability methods and describe the process to extract explanations for student success predictions from a trained model, showcased as feature importance weights.

2.1 Formal Preliminaries

We consider a set of students S enrolled in a course c part of an online educational offering \mathcal{C} . Course c has a prede-

defined weekly schedule consisting of $N = |\mathcal{O}|$ learning objects from a catalog \mathcal{O} . Students enrolled in a course interact with the learning objects included in the course schedule, generating a time-wise clickstream (e.g., a sequence of video plays and pauses, quiz submissions). We denote a clickstream in a course c for a student $s \in S$ as a time series $I_s = \{i_1, \dots, i_{K_s}\}$ with K_s being the total number of interactions of student s in course c . Each interaction $i \in I_s$ is represented by a tuple (t, a, o) , including a timestamp t , an action a (videos: load, play, pause, stop, seek, speed; quiz: submit), and a learning object $o \in \mathcal{O}$ (video, quiz). Given the weekly course schedule, we assume that t_w identifies the time t where the course week $w \in \{0, \dots, W\}$ ends, and that the clickstream of student s generated until the end of the w week can be denoted as $I_s^{t_w}$. We also assume that the course schedule includes one or more assignments per week and that the grade record of student s across course assignments is denoted as $G_s = [g_1, \dots, g_W]$, where $g_w \in G_s$ is the grade student s received on the assignment in week w . In the case of multiple graded assignments for a certain week, we considered the average score of graded assignments for that week and scored non-attempted assignments with 0. We denote as $y_s \in \{0, 1\}$ the success label for student s .

2.2 Data Preprocessing

A significant portion of MOOC students enroll just to watch a few videos or find that the curriculum material is not what they expected and drop out of the course in the first weeks [26, 27]. It follows that it is easy to predict the success labels y_s for this selection of students by simply looking at their initial few weekly assignment grades in G_s . Therefore, optimizing complex deep learning models for predicting student success on early-dropout students is inefficient. Using these complex deep models also leads to less interpretable predictions in comparison with traditional models. For this subset of early-dropout students, traditional models can both achieve a comparable accuracy and still remain interpretable. To identify early-dropout students, we fit a *Logistic Regression* model on the assignment grades of the first two course weeks. The input data is the vector G_s^W , where W is the number of course weeks ($W = 2$ in our experiments) whereas the ground truth is the student success label y_s . Once the model is fitted, we filter out the students that had a predicted probability of course failure $\hat{p}_s > 0.99$. We determine the optimal threshold via a grid search over $\{0.96, 0.97, \dots, 0.999\}$, maximizing the model balanced accuracy. Henceforth, we consider S to be the student population obtained after the early-dropout student filtering.

2.3 Feature Extraction

As an input for our student success prediction models, we consider a set of behavioral features extracted for each student $s \in S$ based on their interactions I_s . We include four feature sets proved to have high predictive power for success prediction in MOOCs [28]. Given the size and variety of the course data considered in our study, we included all features of the four features sets, instead of considering only the specific features identified as important by at least one course [28]. Formally, given interactions I_s generated by students S until a course week w , we create a matrix $H \subset \mathbb{R}^{|S| \times w \times f}$ (i.e, each feature in the feature set is computed per student per week), where $f \in \mathbb{N}$ is the dimensionality of the feature set. We focus on the following behavioral aspects:

¹<http://github.com/epfl-m14ed/evaluating-explainers>

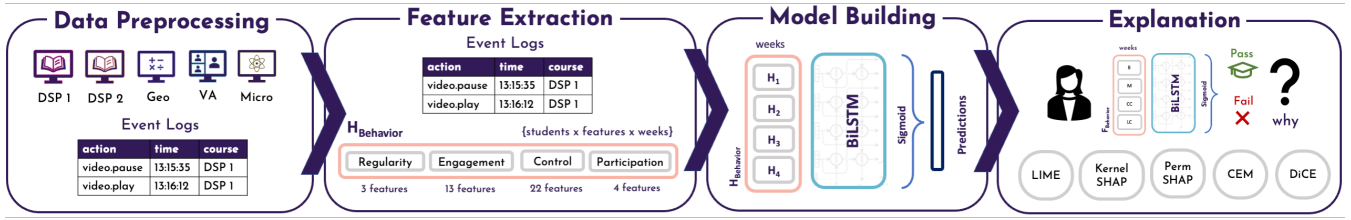


Figure 1: Our experimental pipeline, from data processing to post-hoc explainability methods.

- **Regularity** features (H_1 , shape: $|S| \times w \times 3$) monitor the extent to which a student follows regular study habits [29].
- **Engagement** features (H_2 , shape: $|S| \times w \times 13$) monitor the extent to which a student is engaged in the course [30].
- **Control** features (H_3 , shape: $|S| \times w \times 22$) measure the fine-grained video consumption per student [31].
- **Participation** features (H_4 , shape: $|S| \times w \times 4$) monitor attendance on videos/quizzes based on the schedule [28].

We extract the above features for each student s and concatenate features across sets to obtain the final combined behavioral features h_s per student. The overall matrix of features is defined as $H \in \mathbb{R}^{|S| \times w \times 42}$, with $H = [H_1 \cdot H_2 \cdot H_3 \cdot H_4]$ (\cdot denotes a concatenation). Due to the different scales, we perform a min-max normalization per feature in H (i.e., we scale the feature between 0 and 1 considering all students and weeks for that feature). We elaborate on the most important features later on in the paper as highlighted by the analyses in subsequent experiments (e.g., Table 1).

2.4 Model Building

Given a course c , we are interested in creating a success prediction model that can accurately predict the success label y_s for student s , given the extracted behavioral features h_s . To this end, we rely on a neural architecture based on *Bidirectional LSTMs*, which can provide a good trade-off between effectiveness and efficiency². The model input is represented by H , i.e., the extracted behavior features, having a shape of $|S| \times W \times 42$. NaN values were replaced with the minimum score the student can receive for each respective feature. These features are then fed into a neural architecture composed by two simple yet effective *BiLSTM* layers of size 32 and 64 (loopback of 3) and a *Dense* layer (with Sigmoid activation) having a hidden size of 1. The model outputs the probability the student will pass the course.

2.5 Explanation

Input behavioral features contribute with varying levels of importance to the prediction provided by a success prediction model. We unfortunately cannot examine the importance of these features directly, since deep neural networks act as **black boxes**. Explainability methods can therefore be adopted to approximate the contributions of each feature in H towards the prediction associated with a specific student s . To explore this aspect, we consider five instance-based explainability methods that are popular in the literature and cover different method families [32, 5]. We then compute the feature importance vector for each student s , based on

²Experimental details can be found in Appendices A and B.

each explainability method. Formally, given an explainability method, we denote $e_s \in \mathbb{R}^{w \times 42}$ as the feature importance weights returned by the explainability method for student s . The feature importance weight $e_s[i]$ is a score, comparable across explainability methods, that represents the importance of feature $h_s[i]$ to the model’s individual prediction for student s . The considered explainability methods are described below.

LIME [15] trains a local linear model to explain each individual student instance h_s . To this end, it first generates perturbed instances $h_s^1, h_s^2 \dots h_s^n$ by shifting the feature values of h_s a small amount. These new instances are then passed to the original model to get their associated predictions. Finally, a local interpretable model (e.g., a Support Vector Machine) is trained on the perturbed instances (input) and the corresponding predictions obtained from the original model (labels), weighting perturbed instances by proximity to the original instance. Mathematically, the local model can be expressed with the following equation:

$$\text{LIME}(h_s) = \operatorname{argmin}_{g' \in G'} L(g, g', \pi_{h_s}) + \Omega(g') \quad (1)$$

where h_s is the instance being explained, G' is the family of all possible explanations, L the loss that measures how close the predictions of the explainer g' are to the predictions of the original model g , π_{h_s} is the feature proximity measure, and $\Omega(g')$ represents the complexity of the local model. As LIME returns feature weights $\pi_1 \dots \pi_{|h_s|}$ representing the feature influence on the final decision, we consider these absolute values to be the importance scores e_s , and scale them to the interval $[0, 1]$, where 1 indicates high importance.

KernelSHAP [16] draws inspiration from game-theory based Shapley values (computing feature contributions to the resulting prediction) and LIME (creating locally interpretable models). This SHAP variant uses a specially-weighted local linear regression to estimate SHAP values for any model. Let $x = h_s$ be the student instance being explained. A point x' in the neighborhood of x is generated by first sampling a coalition vector $z \in \mathbb{R}^{|h_s|}$. The coalition vector uses a binary mask to determine which features from x will be kept the same in the new instance x' , and which will be replaced by a random value from the data distribution of that feature in H . Feature importance weights for each new instance x' are calculated using a predefined kernel, after which the local model can be trained. A SHAP explanation is mathematically defined as:

$$g'(z') = \pi_0 + \sum_{h_s=1}^{|h_s|} \pi_{h_s} z'_{h_s} \quad (2)$$

Set	Feature	Description
<i>Regularity</i>	DelayLecture	The average delay in viewing video lectures after they are released to students.
	RegPeakTimeDayHour	Regularity peak based on entropy of the histogram of user’s activity over time.
	RegPeriodicityM1	The extent to which the hourly pattern of user’s activities repeats over days.
<i>Engagement</i>	AvgTimeSessions	The average of users’ time between subsequent sessions.
	NumberOfSessions	The number of unique online sessions the student has participated in.
	RatioClicksWeekendDay	The ratio between the number of clicks in the weekend and the weekdays
	StdTimeSessions	The standard deviation of users’ time between subsequent sessions.
	TotalClicksProblem	The number of clicks that a student has made on problems this week.
	TotalClicksWeekend	The number of clicks that a student has made on the weekends.
	TotalTimeProblem	The total (cumulative) time that a student has spent on problem events.
	TotalTimeVideo	The total (cumulative) time that a student has spent on video events.
StdTimeBetweenSessions	The standard deviation of the time between sessions of each user.	
<i>Control</i>	AvgReplayedWeeklyProp	The ratio of videos replayed over the number of videos available.
	AvgWatchedWeeklyProp	The ratio of videos watched over the number of videos available.
	FrequencyEventLoad	The frequency between every Video.Load action and the following action.
<i>Participation</i>	CompetencyAnticipation	The extent to which the student approaches a quiz provided in subsequent weeks.
	ContentAlignment	The number of videos for that week that have been watched by the student.
	ContentAnticipation	The number of videos covered by the student from those that are in subsequent weeks.
	StudentSpeed	The average time passed between two consecutive attempts for the same quiz.

Table 1: Features used in model explainability analysis. For brevity, we only list the 19 features that have been identified as important by at least one explainability method in our analysis in Section 3.

where g' is the local explainer, $\pi_{h_s} \in \mathbb{R}$ is the SHAP value (feature attribution) of feature h_s , and $z' \in \{0, 1\}^{|h_s|}$ is the coalition binary value. To achieve Shapley compliant weighting, Lundberg et al. [16] propose the SHAP kernel:

$$\pi_{h_s}(z') = \frac{(|h_s| - 1)}{\binom{|h_s|}{|z'|} |z'| (|h_s| - |z'|)} \quad (3)$$

where $|h_s|$ is the maximum coalition size and $|z'|$ is the number of features present in coalition instance z' [5].

SHAP methods directly provide values π_{h_s} representing the feature contribution to the prediction y_s of instance s . To obtain the importance scores e_s , we apply the same transformation as LIME, by taking the absolute values of the SHAP feature attributions and scaling them to the interval $[0, 1]$.

PermutationSHAP (PermSHAP) [16] is very similar to the KernelSHAP formulation, but does not require the tuning of a regularization parameter or a kernel function. We made the decision to include both KernelSHAP and PermSHAP as a form of validation of our comparative evaluation analysis; the distance between two very similar SHAP methods is expected to be smaller than the distance between these SHAP methods and other families of explainability methods. PermSHAP approximates the Shapley values of features by iterating completely through an entire permutation of the features in both forward and reverse directions (antithetic sampling). To extract the feature importance vector e_s , we again consider the absolute values of the SHAP feature attributions and scale to the interval $[0, 1]$.

Contrastive Explanation Method (CEM) [22] identifies which features need to be present (pertinent positives) or which features must be absent (pertinent negatives) in order to

maintain the model prediction y_s for a student s with behavioral features h_s [22]. For our setting, we consider pertinent negatives as they are intuitively more similar, and therefore comparable, to other counterfactual-based explainability methods. For each generated pertinent negative, we calculate the importance score for each feature by multiplying the absolute change from the value in the original instance to the value in the pertinent negative, modeled as the standard deviation (SD) of that feature $\tilde{X}(h_s)$ across all instances used for the experiment $X(h_s)$, as shown in the following formula:

$$\text{CEM}(h_s) = [X(h_s) - \tilde{X}(h_s)] \times SD(h_s) \quad (4)$$

The importance score therefore takes into consideration both the necessary perturbation of the feature as well as the significance of the change relative to the feature range. We normalize the scores in the range $[0, 1]$, such that the resulting feature importance weights e_s can be directly comparable.

Diverse Counterfactual Explanations (DiCE) [20] generates example instances to explain the model prediction as well. However, while CEM describes conditions necessary to keep the prediction unchanged, DiCE describes the smallest possible change to the initial instance that results in a different prediction. In other words, DiCE generates nearest neighbor counterfactual examples by optimizing the loss:

$$\begin{aligned} \text{DiCE}(h_s) = \arg \min_{\mathbf{c}_1, \dots, \mathbf{c}_k} \frac{1}{k} \sum_{i=1}^k y_{\text{loss}}(g(\mathbf{c}_i), y) \\ + \frac{\lambda_1}{k} \sum_{i=1}^k \text{dist}(\mathbf{c}_i, h_s) \\ - \lambda_2 \text{diversity}(\mathbf{c}_1, \dots, \mathbf{c}_k) \end{aligned} \quad (5)$$

where \mathbf{c}_i is a counterfactual example, k is the total number

of examples to be generated, g is the black box ML model, y_{loss} is a metric that minimizes the distance between the prediction g' makes for c_i and the desired outcome y , h_s is the original input with $|h_s|$ input features, and diversity is the Determinantal Point Process (DPP) diversity metric. λ_1 and λ_2 are hyperparameters that balance the three parts of the loss function. The stopping condition is convergence or 5000 time steps per counterfactual. Microsoft’s DiCE library [20] has a built-in function to compute local feature importance scores from the counterfactual instances, scaled in $[0, 1]$. We use them as feature importance weights e_s .

3. EXPERIMENTAL ANALYSIS

We evaluated the explainability methods on five MOOCs. We first explored how feature importance varies across different explainers for one specific course c (RQ1). We then investigated the similarity of the explainability methods across the five courses using distance metrics (RQ2). Finally, we assessed the validity of the explainers using simulated data from a course c with a known underlying prerequisite skill structure (RQ3). In the following sections, we describe the dataset and optimization protocol used for the experiments before explaining each experiment in detail.

3.1 Dataset

Our experiments are based on log data collected from five MOOCs of École Polytechnique Fédérale de Lausanne between 2013 to 2015. We chose the five courses to cover a diverse range of topic, level, and language. Table 2 describes the five courses in detail. We include two subsequent iterations of the same computer science course (*DSP*) with different student populations (French Bachelor students vs. English MSc students). Besides computer science, we also cover courses in the areas of mathematics (*Geomatique*), social sciences (*Villes Africaines*) and engineering (*Micro*). In total, the raw data set contained log data from 75,992 students. After removing the early-dropout students (see Sec. 2.2), 19,805 students remain in the data set. The smallest course contains 452 students, while the largest course contains 5,643 students. Students’ log data consists of fine-grained video (e.g., play, pause, forward, seek) and quiz events (e.g., submit). Interaction data is fully anonymized with regards to student information, respecting participants’ privacy rights.

3.2 Experimental Protocol

For each course $c \in C$, we trained a *BiLSTM* model M_c on features H_c extracted from c . For the optimization, we used batches of size 32, an Adam optimizer with an initial learning rate of 0.001, and a binary crossentropy loss. After an initial grid search³, we selected the same architecture for all models: two *BiLSTM* layers consisting of 64, 32 units and one Dense layer consisting of 1 unit with a Sigmoid activation. As this work is not focused on improving model performance, we did not tune hyperparameters further. Formally, we split the data of each course c into a training data set $S_{train,c}$ (80% of the students) and a test data set $S_{test,c}$ (20% of the students). For each course, we performed a stratified train-test split over students’ pass/fail label. We then trained each model M_c on the training data set $S_{train,c}$ and then predicted student success on the respective test data set $S_{test,c}$. We chose the balanced accuracy

³Grid search is discussed further in Appendix A.

(BAC) as our primary evaluation metric because of the high class imbalance of most of the selected courses.

For the first two experiments (RQ1 and RQ2) we used the student log data collected for the full duration of the course for training and prediction of our models. In the third experiment, we optimized models for different sequence lengths, i.e. using only the log data up to a specific week w of the course (i.e. from week 1 to week w) to predict performance in the assignment of course week w . Additional replication details for model training can be found in Appendix B.

For all experiments, we applied the explainability methods to the predictions of the optimized models M_c . All five methods are instance-based; they compute the feature importance based on the model predictions for a specific instance. Training explainers on the scale of thousands of students across five courses is not feasible due to the computation time required to generate the explanation for one instance (e.g., the counterfactual explainability methods take a computation time of 30 minutes per instance s). Therefore, we determined a representative sampling strategy to pick 100 students from each course c , resulting in explanations for 500 students in total⁴. For the first two experiments (RQ1 and RQ2), we used a uniform sampling strategy to select the representative students $s_{i,c}^r$ for a course c and ensured balance between classes (pass/fail). We first extracted all failing students and ordered them according to the predicted probability of the model $\hat{p}(l_{S_i} = 0)$. We then uniformly sampled 50 failing students from this ordered interval. We repeated this exact same procedure to sample the 50 passing students. This sampling procedure ensures that we include instances where the model is confident and wrong, instances for which the model is unsure, and instances where the model is confident and correct. For the last experiment (RQ3), we used performance in the assignment of a given week w as the binary outcome variable. We then followed exactly the same uniform sampling procedure as for RQ1 and RQ2, ensuring class balance on assignment performance.

3.3 RQ1: Explanations for one course

In a first experiment, we compared the explanations of the instance-based methods for one specific course (*DSP 1*). The *BiLSTM* model M_{DSP1} trained on this course achieved a BAC of 93.9%. We then ran the explainability methods on M_{DSP1} and extracted normalized feature importance scores for 100 representative students of each course.

Figure 2 illustrates the features identified as most important by each explainability method. The heatmaps were computed by averaging importance scores for each feature and week across 100 representative students for *DSP 1* (see Sec. 3.2). To ensure interpretability of Figure 2, we only included the top five features for each method, resulting in 13 distinct features. The description of all the features can be found in Table 1. We used a log scale within the heatmaps, with darker colors indicating higher feature importance.

We observe that the top features cover all the different behavioral aspects included in the feature set: Regularity, Engagement, Control, and Participation. However, some aspects seem to contain more important features. For exam-

⁴Sampling strategy is discussed further in Appendix C.

Title	Identifier	Topic ¹	Level	Language	No. Weeks	No. Students ²	Passing Rate (%)	No. Quizzes
Digital Signal Processing 1	<i>DSP 1</i>	CS	Bsc	French	10	5629	26.8	17
Digital Signal Processing 2	<i>DSP 2</i>	CS	MSc	English	10	4012	23.1	19
Éléments de Géomatique	<i>Geomatique</i>	Math	MSc	French	15	452	45.1	27
Villes Africaines	<i>Villes Africaines</i>	SS	BSc	English	13	5643	9.9	17
Comprendre les Microcontrôleurs	<i>Micro</i>	Eng	BSc	French	13	4069	5.1	18

¹**Topic abbrev.** *Eng*: Engineering; *Math*: Mathematics; *CS*: Computer Science; *SS*: Social Science

²**No. Students** is calculated after filtering out the early-dropout students, as detailed in Sec. 2.2.

Table 2: Detailed information on the five MOOCs included in our experiments.

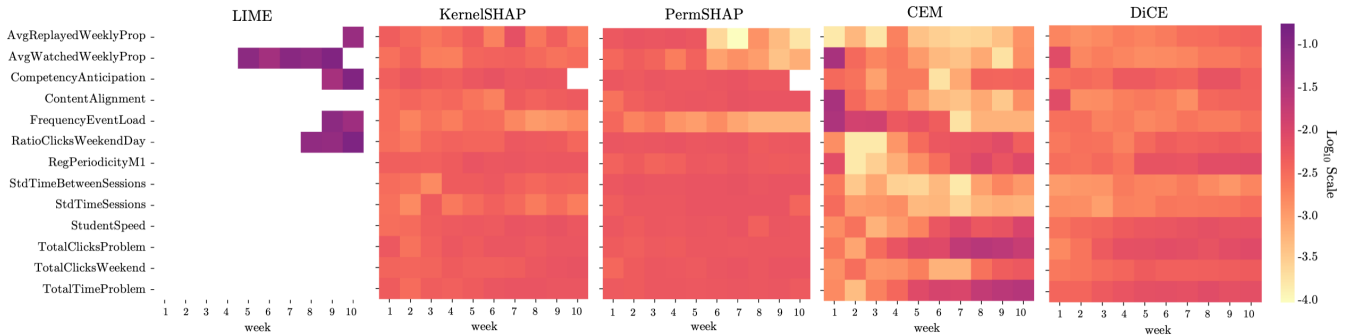


Figure 2: Heatmap of normalized feature importance scores (log scale) across explainability methods for *DSP 1*.

ple, 43% of the Participation features (3 out of 7 features) are in the top five features of at least one method, while this is the case for only 23% (5 out of 22 features) of the Control features. For Regularity and Engagement, 33% and 31% of the features get selected into the top feature set.

We also immediately recognize that the heatmap of LIME looks very different from the heatmaps of all the other methods. LIME assigned high importance scores to a small subset of features and weeks, while all the other explainability methods tend to identify more features and weeks as important, resulting in generally lower importance scores. We also observe that LIME does not consider student behavior in the first weeks of the course important; all importance is placed onto the second half of the course. Moreover, LIME seems to put more emphasis on Control than on the other three aspects: the features related to Control (*AvgReplayedWeeklyProp*, *AvgWatchedWeeklyProp*, *FrequencyEventLoad*) are important from week 5 through week 10, while the features related to Participation (*CompetencyAnticipation*) and Engagement (*RatioClicksWeekendDay*) are important only during the last 2 to 3 weeks of the course.

Interestingly, while CEM and DiCE are both counterfactual methods, their heatmaps look quite different: the feature importance scores of DiCE tend to be more similar to KernelSHAP and PermSHAP than CEM. We note that CEM shows a higher diversity in feature importance scores than the other three methods (KernelSHAP, PermSHAP, and DiCE), for which the importance values seem to be quite equally distributed across the top features. Furthermore, in contrast to all the other explainability methods, CEM seems to also identify features in the first weeks of the course as important (e.g., *AvgWatchedWeeklyProp*, *ContentAlign-*

ment, and *FrequencyEventLoad* in week 1). In contrast to all the other methods, CEM identifies features related to being engaged in quizzes as relevant (*TotalClicksProblem* and *TotalClicksWeekend*). Finally, as expected, the heatmaps of KernelSHAP and PermSHAP look very similar, with only small differences in importance scores.

In summary, while there is some agreement on the top features across explainability methods (the union of the top five features of each method only contains 13 distinct features), we observe differences across methods when it comes to exact importance scores.

3.4 RQ2: Comparing methods across courses

Our second analysis had the goal to *quantitatively* compare the explanations of the different methods across all five courses. Explainability method evaluation is an emerging field; most existing research focused on assessing the quality of explanations [33, 34] with only few works suggesting a quantitative ‘goodness’ score for each explainability method (e.g., [35, 36]). In contrast, we examined the distance between the feature importance scores per explainability method in comparison to each other, instead of individually. We first visualized the similarity of importances across courses using a Principal Component Analysis and then computed Spearman’s Rank-Order Correlation as well as Jensen-Shannon Distance to assess similarity regarding the feature importance ranking as well as their exact values.

Principal Component Analysis (PCA) We performed a PCA on the importance scores for each feature and week (length: $w_c \times h$) separately for each explainability method and course c . Figure 3 shows the results for all explainability methods and courses. Each marker in Figure 3 represents a specific

course, while each color denotes an explainability method.

We observe that the two SHAP methods (KernelSHAP and PermSHAP) cluster together very strongly. This result is expected, as the methodologies of KernelSHAP and PermSHAP are very similar. DiCE feature importances are quite close to the SHAP methods, showing that the three methods have similar notions of feature importance. LIME is quite different from all other methods, with high values on both PCA components. Based on differences in methodology, we would have expected that the difference between the counterfactual methods (DiCE and CEM) and the SHAP methods would be larger than the difference between LIME and the SHAP methods. The most notable takeaway from Figure 3 is that there are clearly identifiable clusters based on explainability method and not on course. It therefore seems that the resulting feature importance scores are mainly influenced by the explainability rather than by the model or data (i.e. the characteristics of the course and students’ data).

Spearman’s Rank-Order Correlation. Often referred to as Spearman’s ρ [37], this metric identifies the rank correlation (statistical dependence between the rankings) between two variables and is defined as the Pearson correlation coefficient between the rankings of two variables. We chose this metric for evaluating explainability methods to highlight the importance of feature ranking order in explanations. To compute Spearman’s Rank-Order Correlation $r_{m_1, m_2, c}$ between two explainability methods m_1 and m_2 on a course c , we first converted the vectors $e_{m_1, s}$ and $e_{m_2, s}$ of feature importance scores (length $w_c \times h$) for each student s into rankings $R(e_{m_1, s})$ and $R(e_{m_2, s})$. We then computed $r_{m_1, m_2, c}^s$ separately for each relevant student s and then averaged over all relevant students to obtain $r_{m_1, m_2, c}$.

Figure 4 illustrates the pairwise similarities between explainability methods using Spearman’s Rank-Order Correlation. Higher values imply stronger correlation between methods. We see similarities between KernelSHAP and PermSHAP

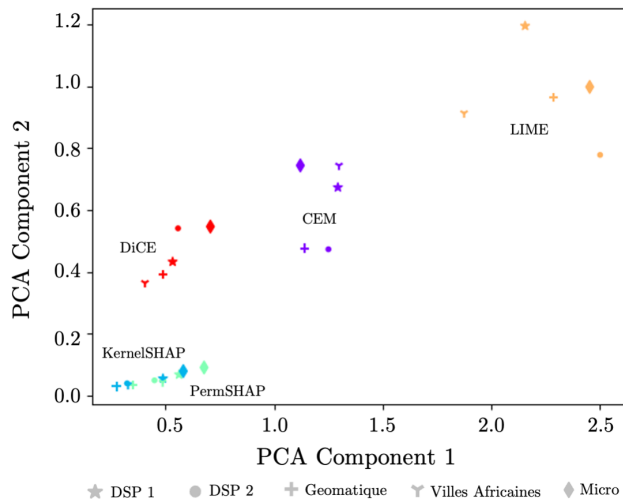


Figure 3: PCA of feature importance scores for five explainability methods across five courses.

prevalent once again as a center square for each course, affirming our intuition that two similar methodologies would result in similar rank-order scores. It can be observed that LIME consistently shows rank-order correlation scores with all other explainability methods. Additionally, for *DSP 1* and to some degree also *Villes Africaines*, DiCE is much closer to KernelSHAP and PermSHAP than to CEM. For *DSP 2* and *Geomatique*, DiCE and CEM are both equally correlated to the SHAP methods, but less correlated among themselves. Finally, the model trained on *Micro* has strong correlations across all explainability methods except LIME.

Jensen-Shannon Distance. We used the Jensen-Shannon distance [38] to compute pairwise distances between exact feature importance score distributions obtained with different explainability methods. The Jensen-Shannon distance is the square root of the Jensen-Shannon divergence, originally based on the Kullback-Leibler divergence with smoothed values. It is also known as the Information Radius (IRad) [39]. To compute the Jensen-Shannon distance $jsd_{m_1, m_2, c}$ between two explainability methods m_1 and m_2 on a course c , we first calculated the distance $jsd_{m_1, m_2, c}^s$ between the feature importance scores (length $w_c \times h$) $e_{m_1, s}$ and $e_{m_2, s}$ separately for each representative student s and then averaged across all representative students to obtain $jsd_{m_1, m_2, c}$.

Figure 5 shows the pairwise distance between explainability methods for all courses using Jensen-Shannon Distance. Larger numbers represent higher dissimilarity. The Jensen-Shannon Distance heatmaps confirm the observations made using Spearman’s Rank-Order Correlation (see Figure 4). Again, LIME consistently has a high distance to all other explainability methods across all courses. As expected, KernelSHAP and PermSHAP have low pairwise distances for all courses. However, when comparing feature importance scores directly instead of using rankings, we observe even less differences between courses. DiCE is closer across all courses to the SHAP methods than CEM. While LIME exhibits the highest distances to all other explainability methods, the explanations of CEM are also far away from all methods.

In summary, the two SHAP methods and DiCE seem to deliver the most similar explanations, while the feature importance scores obtained with CEM and LIME are different from the other explainability methods. More importantly, all our analyses (PCA, Spearman’s Rank-Order Correlation, Jensen Shannon Distance) demonstrate that the choice of explainability method has a much larger influence on the obtained feature importance score than the underlying model and data.

3.5 RQ3: Validation of explanations

The previous experiments delved into mapping the similarities and differences between explainability methods. While our analyses demonstrated that there are clear disparities across method choice, they do not give an indication regarding the ‘goodness’ of the obtained explanations. Recent work discusses traits of ideal explanations [40] and targets metrics to measure explanation quality [36, 41]. However, these metrics tend to be over-specialized to one explainability method over others due to the similarities in their methodologies. The community does not yet have a set of standard metrics for evaluating explainability methods. In our last experiment, we hence use information inherent to

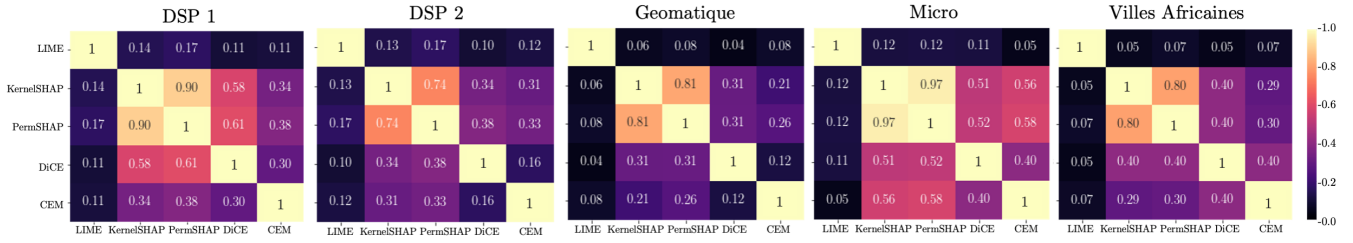


Figure 4: Comparison of feature importance scores across courses using Spearman’s Rank-Order Correlation.

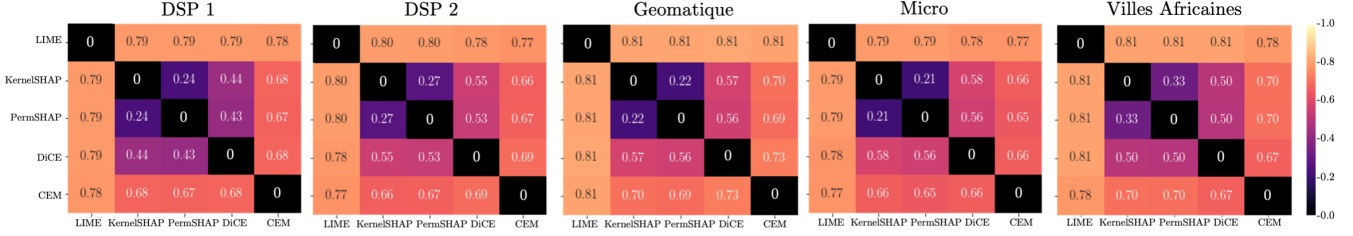


Figure 5: Comparison of feature importance scores across courses using Jensen-Shannon Distance.

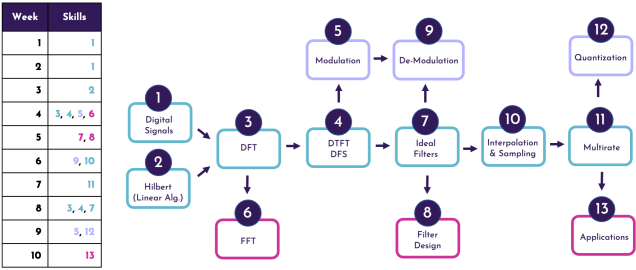


Figure 6: Prerequisite skill structure for DSP 1.

our model’s setting to perform an initial validation of the explanations provided by the different methods.

Specifically, we evaluated the explainability methods on a course with a known underlying skill map and used the prerequisite relationships between weeks of the course as a ground truth for the explanations. Based on the results obtained for the first two research questions (Sections 3.3 and 3.4), we selected one representative method from each methodological group for the analysis, while keeping the observed explanation diversity: PermSHAP (the most widely used SHAP method), CEM (chosen as a representative of counterfactuals for its further disparity from the SHAP methods), and LIME. In terms of courses, we used DSP 1 as a basis for the analysis as the instructor of this course provided us with the skill map derived from the curriculum.

Figure 6 illustrates the underlying skills, their relationships, as well as their mapping to the weeks of the course. The arrows denote the prerequisite relationships, while the numbers denote the unique skills in the order they are introduced in the course. The skills colored in pink (6, 8, and 13) refer to applied skills learned in the course. The middle track refers to core skills learned in the course (colored in blue) and the purple skills at the top (5, 9, and 11) are theory-

based extensions of core material. The skill prerequisite map allows us to analyze the dependencies between the different weeks of the course. For example, in order to understand *Modulation* taught in week 5, students need to already have learned the skills taught in weeks 3 and 4 (*DFT*, *DTFT*, and *DFS*). Intuitively, a model predicting performances in assignments in week 5 would have highly correlated features based on week 4. We assume that these dependencies would logically be uncovered by the explainability method.

We, therefore, adjusted our predictive task: using the optimization protocol and experimental design described in Section 3.2, we aimed at predicting the performance (binary label: below average or above average) of a student s in the assignment of week w based on features extracted from student interactions for weeks 1 to w . Given the prerequisite structure for the course, we ran experiments for $w \in \{5, 9\}$. For each predictive model $M_{DSP1,w}$, we then picked the 100 representative students using uniform sampling and taking into account class balance (see Section 3.2 for a detailed description of the sampling procedure) and applied the selected explainability methods to these representative instances.

Figure 7 shows the features for LIME, PermSHAP, and CEM for the prediction model of week 5. In the heatmap, darker values indicate a higher score. The scores for the heatmap have been computed based on a ranking of features and weeks: for each student s_i^r , we first ranked the features in order of feature importance as determined by the respective explainability method. We then scored each of the top 10 features according to its rank: 10 points for the top feature, 9 points for the second most important feature, and so on. Finally, we averaged the scores for each feature across the 100 representative students s_1^r through s_{100}^r and normalized them. This rank-based scoring allows us to compare explainability methods without having the relative feature importance scores bias the analysis. We only selected features with a score of at least 0.33 in any course week, showing only the top two-thirds of features per method.

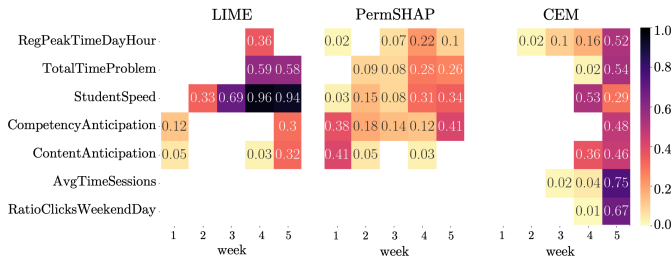


Figure 7: Importance scores for LIME, PermSHAP, and CEM for week 5.

For LIME, we observe that four features have been identified as important for predicting assignment performance in week 5. Two of these features directly relate to student behavior on the week 5 assignment: *TotalTimeProblem* (the total time the student spent solving the assignment) and *StudentSpeed* (the time between consecutive trials of assignments). The high scores of these features for week 5 are thus expected. LIME also assigns high scores to these two assignments based features for week 4. This result is encouraging as week 4 teaches the prerequisite skills of week 5 and therefore, LIME seems to (at least partially) uncover the prerequisite structure of the course.

In PermSHAP results, we see again that the scores are more uniformly distributed across features and weeks. However, we observe again that the assignment-based features (*TotalTimeProblem* and *StudentSpeed*) have comparably high scores for week 4 and therefore PermSHAP also seems to (partially) uncover the prerequisite relationships. Curiously, watching videos and solving quizzes scheduled for subsequent weeks (*CompetencyAnticipation* and *ContentAnticipation*) is also considered important, hinting that being proactive when learning increases learning success.

CEM seems to be able to partially uncover the prerequisite relationship between weeks as well. For week 4, one feature related to assignment behavior (*StudentSpeed*) exhibits a high score. Additionally, watching content of the subsequent week (in this case, week 4’s *ContentAnticipation* for week 5 material) is important for assignment performance. Otherwise, CEM mainly explains performance in the assignment of week 5 with student behavior in week 5: besides the assignment-related features, the students’ actions in the inference week are considered important.

Figure 8 shows the importance scores for LIME, PermSHAP, and CEM for the prediction model of week 9. Again, darker colors in the heatmaps indicate higher scores. The scores in the heatmaps were computed using the same ranking-based procedure as for Figure 7. In week 9, Figure 6 indicates that weeks 5 and 6 cover the prerequisite skills. From Figure 8, we observe that LIME does not seem to be able to capture this prerequisite relationship. The top scores for LIME appear in week 9 itself for an assignment-based feature (*StudentSpeed*) as well as for video control behavior (*AvgReplayedWeeklyProp*, which computes the relative number of video replays). Furthermore, *StudentSpeed* seems to be generally important also in the weeks just before the predicted week (weeks 7 and 8). For PermSHAP, we again obtain

a more equal distribution, with only the two assignment-related features (*TotalTimeProblem* and *StudentSpeed*) in weeks 8 and 9 showing relatively higher scores. Furthermore, PermSHAP also assigns relatively higher importance to these features for week 6, which is a prerequisite for week 9. For CEM, we again observe that mainly student behavior in the actual week, i.e. week 9, seems to explain assignment performance. Only one feature (*TotalTimeVideo*) shows medium importance for weeks 5 and 6.

In summary, all the evaluated methods were able to (partially) detect the prerequisite relationship between week 4 and week 5. For week 9, detecting the prerequisite structure proved to be difficult; results differed between methods. However, we should take into account that none of the features of the feature set directly measure student performance and therefore, the generated explanations rely on behavioral features only. It appears that recent and actual behavior is a much stronger indicator for performance than past behavior.

4. DISCUSSION AND CONCLUSION

Explainability methods allow us to interpret a deep model in a way that is understandable not only to machine learning experts, but also end-users of educational environments, including instructors tailoring course designs and students the model is predicting on [6, 7]. In this paper, we aimed to understand explainers’ behaviour and the ways in which they differ for the task of student success prediction.

Our results demonstrate that all explainability methods can derive interpretable motivations behind student success predictions, confirming the similar yet coherent observations made by [8] for the knowledge tracing field. However, while there was some agreement regarding the top features across the five explainability methods, key differences across methods emerged when we considered the exact importance scores (RQ1). We observed substantial similarities between KernelSHAP, PermSHAP, CEM and DiCE with regards to the top ranked feature-weeks. Conversely, LIME only ranked very few features as important, and these less important feature similarities made the other explainability methods appear closer to each other. Overall, looking beyond top ranked features, we noted considerable differences in feature importances across explainability methods. Interestingly, LIME-detected features are more in line with the features marked as important by Random Forests in [28], still in a MOOC context. This observation further demonstrates the generalizability of the features’ predictive power even among very different experimental settings.

In a subsequent experiment, we compared the different explainability methods across five MOOCs. Our findings indicate that the choice of explainability method has a much larger influence on the obtained feature importance score than the underlying model and data (RQ2). With distance (Jensen-Shannon distance) and ranking-based metrics (Spearman’s Rank-Order Correlation), we uncovered that LIME is farthest from the other explainability methods. The sparsity of LIME-detected important features was also observed by [13], where the conciseness of LIME explanations supported integration in visual dashboards for student advising. We also detected a close relationship between KernelSHAP and PermSHAP, which strongly validates our evaluation strat-

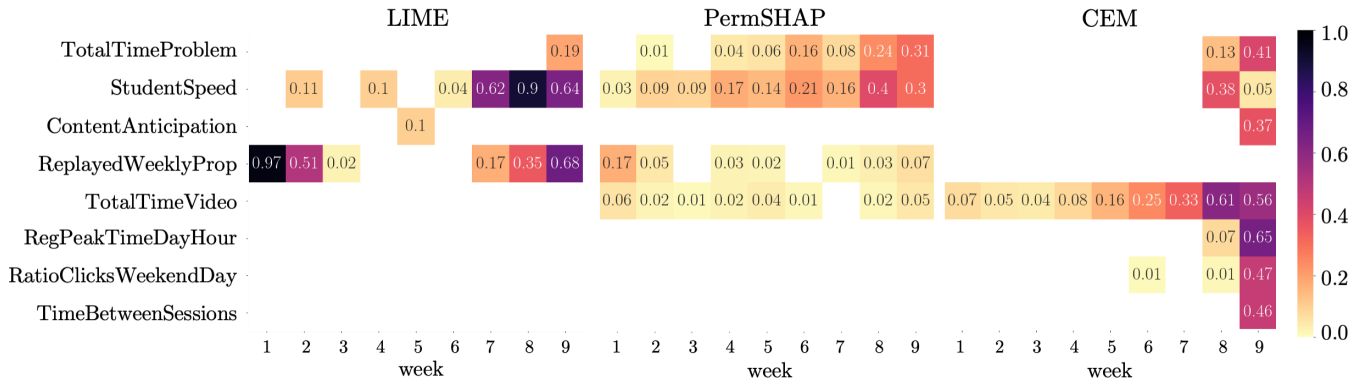


Figure 8: Importance scores for LIME, PermSHAP, and CEM for week 9.

egy. Using PCA, we identified clear clusters of explanations by explainability method and not by the course the model was trained upon, suggesting that an explainability method might be prone to mark specific features as important regardless of the model (and the course).

Our analyses also confirmed that all the evaluated methods were able to (partially) detect the prerequisite relationship between weeks, while relying on behavioral features only (RQ3). Our experimental design was inspired by [11]’s work on predicting effectiveness of interventions for wheel-spinning students by simulating prerequisite relationships. While we have no way to examine the true underlying feature importances of our week n assignment performance prediction model, we intuit that a student’s prerequisite week performance should be important to predicting their performance in week n . We observed that the three families of methods (LIME, SHAP, and counterfactual) were able to partially capture the prerequisite relationship in week 5, but struggled to capture the prerequisite relationships in week 9. While there were few similarities in the top ranked features, each method found different groups of feature-weeks as most important for the same models. Our results indicate that recent and current behavior is more important than past behavior, implying that proximity of behavioural features correlates strongly with their perceived importance. A limitation is that the prerequisite relationships we deem important might not actually be used as the true features of the model since our feature set included only features that examined student behavior and not direct performance.

Our results indicate that there are noteworthy differences in generated explanations for student performance prediction models. However, our analyses also show that these explanations often recognize prerequisite-based relationships between features. That being said, our study still has several limitations that warrant future research, including our focus on a singular downstream task (student success prediction), specific modality of dataset (MOOCs), choice of model architecture (*BiLSTM*), and lack of assessment of the obtained explanations’ impact in the real world. First, extending our experiments beyond success prediction to a multi-task analysis (e.g., dropout prediction) across multiple modalities (e.g., flipped classrooms, intelligent tutoring systems) would allow us to build stronger intuitions about explain-

ability method differences. Second, extending our black-box *BiLSTM* model architecture to multiple traditional and deep machine learning architectures could examine whether certain explainability methods have stronger explanation affinity to different predictors. Choosing transparent shallow architectures instead of black-boxes could also allow us to validate our results against ground truth feature importances. Third, further research should be conducted to check which explanations (and explainability methods) lead to interventions that better improve learning outcomes. It follows that an assessment of the obtained explanations should be carried out involving educators. Finally, the disagreement of our selected explainability methods motivate an extension in an ensemble expert-weighting scheme, which might have merit in closely estimating the true feature importances.

Explainability in educational deep learning models can lead to better-informed personalized interventions [2, 42], curriculum personalization, and informed course design. If we were considering global interventions (as it might be too resource intensive to perform interventions on each student individually), we could take the mean feature importance vector over all students and try interventions in the order of the scores of this mean vector. If we were only able to intervene on k features due to resource constraints, Spearman’s rank-order metric could also be modified to include the size of the intersection between the features with the top k scores. However, it is important to note that when the model explanations are biased by explainability method and do not accurately reflect the inner workings of the model, the impact of incorrect predictions are further exacerbated by teachers and students’ misplaced confidence in the model’s justification. **We implore data scientists to not take the choice of explainability method lightly as it does have a significant impact on model interpretation, and instead urge the community to (1) carefully select an appropriate explainability method based on a downstream task and (2) keep potential biases of the explainer in mind when analyzing interpretability results.** Overall, our work contributes to ongoing research in explainable analytics and to the generalization of theories and patterns in success prediction.

5. ACKNOWLEDGMENTS

We thank Professor Paolo Pardon, Valentin Hartmann, and Natasa Tagasovska for their expertise and support.

6. REFERENCES

- [1] Ali Shariq Imran, Fisnik Dalipi, and Zenun Kastrati. Predicting student dropout in a MOOC: An evaluation of a deep neural network model. In *5th International Conference on Computing and Artificial Intelligence*, pages 190–195, 2019.
- [2] Wanli Xing and Dongping Du. Dropout prediction in MOOCs: Using deep learning for personalized intervention. *Journal of Educational Computing Research*, 57(3):547–570, 2019.
- [3] Chris Piech, Jonathan Bassen, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas J Guibas, and Jascha Sohl-Dickstein. Deep Knowledge Tracing. *Advances in Neural Information Processing Systems*, 28, 2015.
- [4] Ghodai Abdelrahman and Qing Wang. Knowledge tracing with sequential key-value memory networks. In *42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 175–184, 2019.
- [5] Christoph Molnar. *Interpretable Machine Learning*. 2nd edition, 2022.
- [6] Mary E Webb, Andrew Fluck, Johannes Magenheimer, Joyce Malyn-Smith, Juliet Waters, Michelle Deschênes, and Jason Zagami. Machine learning for human learners: opportunities, issues, tensions and threats. *Educational Technology Research and Development*, 69(4):2109–2130, 2021.
- [7] Cristina Conati, Kaska Porayska-Pomsta, and Manolis Mavrikis. Alin education needs interpretable machine learning: Lessons from open learner modelling. 2018.
- [8] Yu Lu, Deliang Wang, Qinggang Meng, and Penghe Chen. Towards interpretable deep learning models for knowledge tracing. In *International Conference on Artificial Intelligence in Education*, pages 185–190. Springer, 2020.
- [9] Khan Md. Hasib, Farhana Rahman, Rashik Hasnat, and Md. Golam Rabiul Alam. A machine learning and explainable AI approach for predicting secondary school student performance. In *IEEE 12th Annual Computing and Communication Workshop and Conference*, pages 0399–0405, 2022.
- [10] Máté Baranyi, Marcell Nagy, and Roland Molontay. Interpretable deep learning for university dropout prediction. In *21st Annual Conference on Information Technology Education*, pages 13–19, 2020.
- [11] Tong Mu, Andrea Jetten, and Emma Brunskill. Towards suggesting actionable interventions for wheel-spinning students. *International Educational Data Mining Society*, 2020.
- [12] Alexandra Vultureanu-Albiși and Costin Bădică. Improving students’ performance by interpretable explanations using ensemble tree-based approaches. In *IEEE 15th International Symposium on Applied Computational Intelligence and Informatics*, pages 215–220. IEEE, 2021.
- [13] Hanne Scheers and Tinne De Laet. Interactive and explainable advising dashboard opens the black box of student success prediction. In *European Conference on Technology Enhanced Learning*, pages 52–66. Springer, 2021.
- [14] Bo Pei and Wanli Xing. An interpretable pipeline for identifying at-risk students. *Journal of Educational Computing Research*, page 07356331211038168, 2021.
- [15] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. ”Why Should I Trust You?”: Explaining the predictions of any classifier. In *22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144, 2016.
- [16] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in Neural Information Processing Systems*, 30, 2017.
- [17] Gajendra Jung Katuwal and Robert Chen. Machine learning model interpretability for precision medicine. *arXiv preprint arXiv:1610.09045*, 2016.
- [18] André Ferreira, Sara C Madeira, Marta Gromicho, Mamede de Carvalho, Susana Vinga, and Alexandra M Carvalho. Predictive medicine using interpretable recurrent neural networks. In *International Conference on Pattern Recognition*, pages 187–202. Springer, 2021.
- [19] Alex Gramegna and Paolo Giudici. Shap and lime: An evaluation of discriminative power in credit risk. *Frontiers in Artificial Intelligence*, 4, 2021.
- [20] Ramaravind K Mothilal, Amit Sharma, and Chenhao Tan. Explaining machine learning classifiers through diverse counterfactual explanations. In *Conference on Fairness, Accountability, and Transparency*, pages 607–617, 2020.
- [21] Janis Klaise, Arnaud Van Looveren, Giovanni Vacanti, and Alexandru Coca. Alibi explain: algorithms for explaining machine learning models. *Journal of Machine Learning Research*, 22(181):1–7, 2021.
- [22] Amit Dhurandhar, Pin-Yu Chen, Ronny Luss, Chun-Chen Tu, Paishun Ting, Karthikeyan Shanmugam, and Payel Das. Explanations based on the missing: Towards contrastive explanations with pertinent negatives. *Advances in Neural Information Processing Systems*, 31, 2018.
- [23] Yash Goyal, Ziyang Wu, Jan Ernst, Dhruv Batra, Devi Parikh, and Stefan Lee. Counterfactual visual explanations. In *International Conference on Machine Learning*, pages 2376–2384. PMLR, 2019.
- [24] Martin Pawelczyk, Klaus Broelemann, and Gjergji Kasneci. Learning model-agnostic counterfactual explanations for tabular data. In *The Web Conference*, pages 3126–3132, 2020.
- [25] David Martens and Foster Provost. Explaining data-driven document classifications. *Management Information Systems Quarterly*, 38(1):73–100, 2014.
- [26] Daniel FO Onah, Jane Sinclair, and Russell Boyatt. Dropout rates of massive open online courses: behavioural patterns. *6th International Conference on Education and New Learning Technologies*, 1:5825–5834, 2014.
- [27] Joselyn Goopio and Catherine Cheung. The mooc dropout phenomenon and retention strategies. *Journal of Teaching in Travel & Tourism*, 21(2):177–197, 2021.
- [28] Mirko Marras, Julien Tuang Tu Vignoud, and Tanja Kaser. Can feature predictive power generalize? benchmarking early predictors of student success across flipped and online courses. In *14th International Conference on Educational Data Mining*, pages 150–160, 2021.
- [29] Mina Shirvani Boroujeni, Kshitij Sharma, Lukasz

- Kidziński, Lorenzo Lucignano, and Pierre Dillenbourg. How to quantify student’s regularity? In *European conference on technology enhanced learning*, pages 277–291. Springer, 2016.
- [30] Fu Chen and Ying Cui. Utilizing student time series behaviour in learning management systems for early prediction of course performance. *Journal of Learning Analytics*, 7(2):1–17, 2020.
- [31] Sébastien Lallé and Cristina Conati. A data-driven student model to provide adaptive support during video watching across MOOCs. In *International Conference on Artificial Intelligence in Education*, pages 282–295. Springer, 2020.
- [32] Pantelis Linardatos, Vasilis Papastefanopoulos, and Sotiris Kotsiantis. Explainable AI: A review of machine learning interpretability methods. *Entropy*, 23(1):18, 2020.
- [33] Kacper Sokol and Peter Flach. Explainability fact sheets: a framework for systematic assessment of explainable approaches. In *Conference on Fairness, Accountability, and Transparency*, pages 56–67, 2020.
- [34] David B Leake. *Evaluating explanations: A content theory*. Psychology Press, 2014.
- [35] An-phi Nguyen and María Rodríguez Martínez. On quantitative aspects of model interpretability. *arXiv preprint arXiv:2007.07584*, 2020.
- [36] Jianlong Zhou, Amir H Gandomi, Fang Chen, and Andreas Holzinger. Evaluating the quality of machine learning explanations: A survey on methods and metrics. *Electronics*, 10(5):593, 2021.
- [37] Charles Spearman. The proof and measurement of association between two things. 1961.
- [38] Jianhua Lin. Divergence measures based on the shannon entropy. *IEEE Transactions on Information theory*, 37(1):145–151, 1991.
- [39] Christopher Manning and Hinrich Schütze. *Foundations of statistical natural language processing*. MIT press, 1999.
- [40] Diogo V Carvalho, Eduardo M Pereira, and Jaime S Cardoso. Machine learning interpretability: A survey on methods and metrics. *Electronics*, 8(8):832, 2019.
- [41] Yunzhe Jia, Eibe Frank, Bernhard Pfahringer, Albert Bifet, and Nick Lim. Studying and exploiting the relationship between model accuracy and explanation quality. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 699–714. Springer, 2021.
- [42] Amir-Hossein Karimi, Bernhard Schölkopf, and Isabel Valera. Algorithmic recourse: from counterfactual explanations to interventions. In *ACM Conference on Fairness, Accountability, and Transparency*, pages 353–362, 2021.
- [43] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. {TensorFlow}: A system for {Large-Scale} machine learning. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, pages 265–283, 2016.

APPENDIX

A. MODEL ARCHITECTURE

We experimented with traditional machine learning models (e.g., Support Vector Machines, Logistic Regression, Random Forest) and deep-learning models (e.g., Fully-Connected Networks, RNNs, LSTMs, CNNs, and BiLSTMs), and found that BiLSTM models perform best against the other baselines for our use case. To determine the optimal model architecture, we evaluate validation set performance on the course *DSP 1* as it is used in all three RQs. BiLSTMs have a 26.8% average increase in balanced accuracy over traditional machine learning methods. For the BiLSTM architecture grid search, we examined the following layer settings {32, 64, 128, 256, 32-32, 64-32, 128-32, 128-64, 128-128, 256-128, 64-64-32, 128-64-32, 64-64-64-32-32} before determining 64-32 performed best in balanced accuracy for *DSP 1*. We used the Tensorflow library to train our models [43].

B. MODEL TRAINING

Model training took approximately 35 minutes per model on an Intel Xeon E5-2680 CPU with 12 cores, 24 threads, and 256 GB RAM. Each model was trained for 15 epochs, and the best performing model checkpoint was saved. The five models’ performance metrics are showcased in Table 3.

C. SAMPLING STRATEGY

We experimented with several strategies to extract an appropriate representative sample including the greedy algorithm SP-LIME [15], random sampling, two sets of extreme students (those which the model predicts very well on and very badly on), and uniform sampling. We determined that our uniform sampling approach was the most fair with respect to the variable class imbalance between courses.

Identifier	Accuracy	Balanced Accuracy	F-1 Score
<i>DSP 1</i>	99.3	97.4	99.6
<i>DSP 2</i>	99.1	93.5	99.5
<i>Geomatique</i>	97.7	96.2	98.7
<i>Villes Africaines</i>	98.4	95.5	99.1
<i>Micro</i>	89.5	90.9	90

Table 3: Performance of the BiLSTMs trained on the five MOOCs included in Section 3.

Addressing Competing Objectives in Allocating Funds to Scholarships and Need-based Financial Aid

Vinhthuy Phan[†]
The University of Memphis
Memphis, Tennessee, USA
vphan@memphis.edu

Laura Wright
The University of Memphis
Memphis, Tennessee, USA
lmstockbridge@gmail.com

Bridgette Decent
The University of Memphis
Memphis, Tennessee, USA
bdecent@memphis.edu

ABSTRACT

A strategy for allocating merit-based awards and need-based aid is critical to a university. Such a strategy, however, must address multiple, sometimes competing objectives. We introduce an approach that couples a gradient boosting classifier for predicting outcomes from an allocation strategy with a local search optimization algorithm, which optimizes strategies based on their expected outcomes. Unlike most existing approaches that focus strictly on allocating merit-based awards, ours optimizes simultaneously the allocation of both merit-based awards and need-based aid. Further, the multi-objective optimization lets users experiment with different combinations of institution-centric and student-centric objectives to deliver outcomes that suit desired goals. With this approach, we identify multiple allocation strategies that would yield higher enrollment, revenue, or students' affordability and access to higher education than the University's existing strategy. In particular, one strategy suggests that with moderate changes to the current funding structure the University can increase students' access to higher education by more than 100%, while still maintaining a similar level of enrollment and revenue.

Keywords

institutional data analytics, financial aid optimization, scholarship distribution, enrollment management

1. INTRODUCTION

Public universities in the United States (US) create recruitment policies amidst decreased state funding and increased costs of attendance. Decreased funding was predicted as a result of state deficits [17]. This is now evident in data showing falling state appropriations as a percent of total revenue for public 4-year institutions from academic year 2008-09

(23.8%) to academic year 2018-19 (16.5%) [12]. Decreased state funding necessitates a greater reliance on tuition and fee revenue from enrolled students.

From academic year 2011-12 through 2016-17 tuition and fees increased by 9% at public 4-year institutions, making college too costly for an increasing proportion of students (especially those from low- and lower-middle-income families) [3, 9, 26]. Students are being priced out of college or choosing to take out loans and/or work while pursuing a course of study [25]. This rising cost has resulted in a national college debt crisis in which 57% of debtors with federal student loan debt owe up to \$20,000 [3].

The combination of decreased funding and increased costs of attendance have implications for enrollment management activities. An increased reliance on tuition and fee revenue means that setting policies surrounding student recruitment efforts and accurately forecasting their outcomes is increasingly important to universities' income and budget projections [2, 38]. However, these recruitment efforts are hindered by the increased prices charged to students. Higher prices reduce enrollment rates particularly for low-income students in public institutions [22]. The use of financial aid as an enrollment management tool to counteract these increased prices is well-established [15]. Further, the positive impact of both federal and state need-based financial aid programs on students from low-income families has also been explored [3, 8, 9]. With increasing costs of attendance leading to crises of affordability and access, universities should explore the role of need-based aid in their enrollment management strategies.

The call to optimize enrollment management activities with the consideration of access and affordability is one that appears difficult to achieve. However, the student and financial data needed for such predictions and optimizations are stored in universities' existing data systems [31]. Despite having access to this data, universities are not quick to leverage it themselves perhaps due to untrained personnel or poor data infrastructure [31, 32]. Instead, the data and these analyses are outsourced to external consulting firms who organize and process the data to be used in proprietary models [16]. With this type of industry environment there is limited published work to demonstrate how machine learning prediction and optimization can be used in tandem to inform university recruitment policies.

[†]corresponding author

V. Phan, L. Wright, and B. Decent. Addressing competing objectives in allocating funds to scholarships and need-based financial aid. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 110–121, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853028>

Existing work on financial aid optimization has been limited in two regards: (1) the optimization only takes enrollment as the sole objective and (2) the resulting strategy is for merit-based aid only [4, 28]. The limiting of the optimization to enrollment is short-sighted in that universities are targeting multiple objectives that may not move with enrollment. Additionally, when only considering merit-based awards, student-centric outcomes like access and affordability are neglected despite their increasing importance in the dialogue surrounding higher education. Our work seeks to address these shortcomings.

Our approach consists of an interactive feedback loop between two core components: a gradient-boosting classifier and *auto-start* local search optimizer. First, given a current strategy for award and aid allocation (together with many other features), the classifier is used to predict enrollment, revenue, the affordability and accessibility of higher education, and other outcomes. These predicted outcomes are then used by an optimizer to revise the current strategy and suggest a possibly better one. The interaction between the classifier and optimizer forms a feedback loop that is the basis of the optimization. This loop aims to identify optimal strategies for allocating awards and aid with expected outcomes specified by the University. The novelty of this work includes (1) the allocation both merit-based awards and need-based aid in the optimization process; (2) the inclusion of the affordability and accessibility of higher education in the optimization of allocation strategies; and (3) an *auto-start* local search optimization that allows users to target institution-centric and student-centric objectives in a flexible way to produce desirable outcomes.

2. RELATED WORK

The following describes research on enrollment prediction, scholarship allocation, and financial aid optimization. There is significant work on predicting enrollment, but far less with regard to optimization and allocation of scholarships. Most authors looked at only one of these issues and most work has been primarily concerned with merit-based aid alone.

Predicting Enrollment. Universities' increased reliance on tuition and fee revenue has increased pressure on their limited resources to accurately predict and increase enrollment (i.e. yield). Recruitment counselors need to allocate their limited time to contact students "on-the-fence" about their enrollment decisions [10]. This business need led to the literature on enrollment prediction at the individual applicant level using data mining and machine learning techniques.

There is a host of research that relies on logistic regression techniques to create these individual enrollment probabilities. A 2002 paper by DesJardins used a logistic regression to reveal the enrollment probabilities of applicants before any financial aid offers had been made. The resulting model, which relied heavily on demographic features, had a correct classification rate of approximately 67% [10]. In 2006, Goenner and Pauls used a logistic regression to predict the enrollment of students who had inquired about the university. The resulting model was 89% accurate at out-of-sample predictions [14]. Similarly, a 2014 paper by Sugrue generated a logistic regression to predict enrollment yield [34].

In another vein of literature, other prediction techniques are explored. Two related works explore the effectiveness of neural network models in predicting individual enrollment probabilities for applicants. A 1998 paper by Walczak revealed that a backpropagation neural network model could result in a 56% decrease in recruitment counselor caseloads due to its accuracy [39]. In a follow-up study, Walczak and Sincich explicitly compare the performance of neural network models to that of logistic regression and find that neural networks produce better results [40]. Chang similarly explores the efficacy of various prediction techniques, specifying models for logistic regression, neural networks, and classification and regression tree (CR&T). The work showed that both neural networks and CR&T outperformed logistic regression when judged based on prediction accuracy [6].

Scholarship Disbursement/Allocation. Not all scholarship policies employed by universities are rule-based. In these cases, the business process of selecting the applicants to receive an award may be ineffective and inefficient. Literature has shown that data mining techniques can be effective in creating rule-based scholarship allocation policies to help reduce business process inefficiencies. In 2019, Rohman et al. illustrated how an ID3 decision tree algorithm could generate rules to select the scholarship applicants most likely to be awarded a scholarship. This general rule allowed for the efficient identification of applicants so that offers could be made [27]. Alhassan and Lawal similarly used a tree-based data mining classification technique to determine a generic rule for scholarship disbursement. They found the technique to be effective and efficient [1].

Optimization of financial aid policies. Mathematical programming models are effective tools for generating and evaluating financial aid strategies. Spaulding and Olswang use discriminant analysis to test the efficacy of various aid strategies [33]. Linear programming was used by Sugrue et al in 2006 as an aid decision tool where the goal was to maximize net revenue with budget, average Scholastic Assessment Test (SAT) scores, recruitment pools, and enrollment targets as constraints [37]. In later work, Sugrue again employed a linear programming approach to optimize the quality of the incoming class with estimates of yield rates being derived from previous years' yield rates [35].

More recent research has incorporated enrollment prediction models with optimization techniques to recommend financial aid strategies. In 2015 Sarafraz et al used a neural network model to predict enrollment and then employed a genetic algorithm to find a scholarship strategy that maximized total enrollment [28]. In 2019 Sugrue used data from the University of Miami to develop a merit-based aid allocation model that predicted enrollment via logistic regression and maximized the quality of the incoming class via a linear programming model [36]. In 2020 Aulck et al tested a group of machine learning tools to predict the enrollment decisions of admitted, domestic nonresident first-time students and then used those results in a genetic algorithm to recommend an optimal disbursement strategy for a domestic non-resident merit scholarship that would maximize enrollment [4].

None of these previous works have considered merit-based award and need-based aid strategies simultaneously as our

paper proposes to do. Further, this joint consideration demands the consideration of a what an appropriate optimization objective should be given that the recruitment goals of merit-based awards and need-based aid differ in some regards. Though the joint consideration of merit and need strategies in one optimization problem does complicate the process, a multi-objective approach needs to be researched since these aid strategies compete for the same limited budget resources.

Local-search optimization. Local search is a heuristic method for solving computationally hard problems. A local search algorithm starts with an initial candidate solution and iteratively moves to a neighbor solution in hopes of finding better and better candidate solutions. The algorithm stops when it cannot find a neighbor solution that is better than the current candidate solution. For local search to work, a *neighbor relation* must be defined so that from an arbitrary candidate solution, a neighbor solution can be generated.

Stochastic hill climbing [20] is a fast local search method because it greedily moves from one candidate solution to a better one. It is similar to a popular method, stochastic gradient descent [19], but it is faster because it does not need to estimate the gradient of the objective function. The main disadvantage of stochastic hill climbing is that it is often stuck in locally optimal solutions.

Two popular methods that can find globally optimal solutions are simulated annealing [21] and genetic algorithms [41]. In a number of applications, genetic algorithms produced slightly better solutions than simulated annealing [30]. Nevertheless, simulated annealing and its variant, simulated quenching [18], seem to suit our technical approach better than genetic algorithm because it is not obvious to us how we can meaningfully apply the genetic algorithm’s crossover operator to allocation strategies.

3. METHODS

3.1 Overview of the problem and challenges

Merit-based awards and need-based aid. At the University, a certain amount of financial award or aid is offered to each applicant based on his or her profile on merit (academic performance) and need. Merit-based award eligibility is calculated based on a combination of standardized test scores (i.e. American College Testing (ACT) and Scholastic Assessment Test (SAT)) and high school GPA. Need-based aid eligibility is determined by expected family contribution from the Free Application for Federal Student Aid (FAFSA) and then adjusted in an ad hoc manner based on items such as cost of attendance and how much federal aid and other merit-based awards are promised to students. A student may receive both merit awards and financial aid. The allocation strategy of merit awards and aid strives to be fair in that two students with the same residency and academic performance profile will receive the same merit offer and two students with the same residency and need profile will receive the same aid offer. These offers are made in a guaranteed fashion in that each admitted applicant will be offered a merit-based award and/or need-based aid so long as requirements are met.

Allocation strategy of awards and aid. In the context of this work, an allocation strategy consists of a merit-based allo-

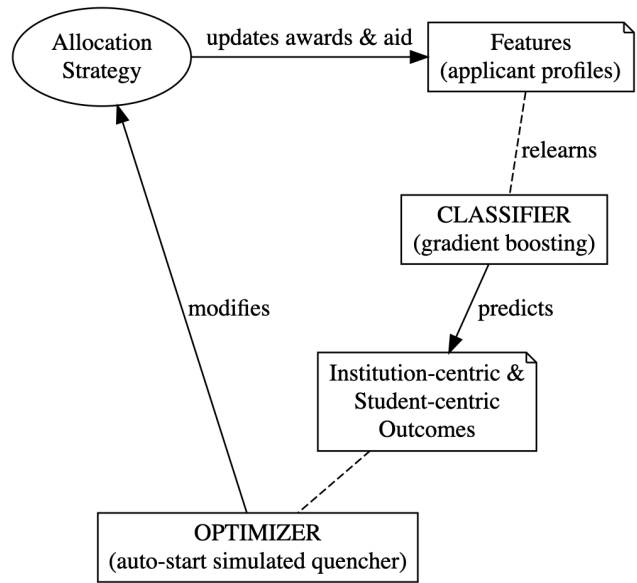


Figure 1: Overview of the technical approach

cation strategy and a need-based allocation strategy. Each allocation strategy consists of (1) a list of buckets into which students are placed, and (2) how much financial award or aid students in each bucket will receive.

An allocation strategy has an important impact on the budget of the University because it affects enrollment in complex ways. Enrollment affects revenue. Revenue affects the number and amounts of awards and aid the University can give to its students. And awards and aid directly affect the applicants’ decision to enroll at the University. Increasing awards and aid can increase enrollment, but may reduce revenue, which in turn limits the University’s ability to increase awards and aid.

Our mission. Our team was tasked with revising the University’s current way of offering merit-based awards and need-based aid. Although the current system is fair, as it should be, the Financial Aid Office has thought that due to multiple reasons the current strategy might not be optimal. We were asked to generate a revised strategy that uniformly impacts the domestic student body in a way that improves multiple, possibly conflicting objectives such as increasing enrollment, increasing revenue, increasing student performance profiles, and making higher education more accessible and affordable.

Technical approach. An overview of our approach is depicted in Figure 1. The approach utilizes a feedback loop that continually revises a feature set from which a classifier learns to predict various outcomes. In contrast to the traditional setting where features stay fixed, the features in our model are continually revised in this feedback loop. This is possible due to the fact that the University can change award and aid offer amounts, which is part of the feature set used to predict enrollment, revenue and other outcomes. The feedback loop is an iterative interaction between the classifier, which predicts expected outcomes, and an optimizer, which evaluates the outcomes and suggests a revised strategy with

possibly better outcomes. In each iteration of this interaction, depicted in Figure 1, the current allocation strategy dictates offer amounts for awards and aid. This changes the features, which leads the classifier to relearn and re-predict multiple outcomes (e.g. enrollment, revenue, student affordability and accessibility, etc.). This allows the *auto-start* local-search optimizer to revise and possibly improve the current strategy; and so on and so forth.

3.2 Data Collection and Cleaning

Data for this research is all first-time freshmen (FTF) domestic admitted applicants to the University in the Fall 2020. The data on students is compiled from the University’s admission application, the FAFSA, and the student information system. Only the students who filed a FAFSA (making them eligible for need-based aid) are included resulting 7,564 observations which is approximately 67% of the total number of domestic FTF students for that term. Data from the admission application includes admission test scores, high school GPA, location/residency, and intended major. Data from the FAFSA includes expected family contribution and family income. Data from the student information system includes whether or not the student had previously enrolled at the institution as a dual enrollment student, aid offer amounts for various categories of aid, and an enrollment indicator. Of the 7,564 students 2,340 chose to enroll (approximately 31%). No demographic variables are used in the study so as to avoid bias in recommending aid strategies to the University.

Financial data points are expected full time tuition, merit-based award offers, need-based aid offers, actual and estimated Pell grant offers, and all other aid offers before loans. Estimated Pell grant offers are included as the University will make offers prior to actual Pell amounts being known. These are estimated based on cost of attendance and expected family contribution using the Pell schedules released each year with the assumption that students will enroll full time. Each of the aid offer features enters the predictive model independently and in a total financial aid variable. These offer amounts are not manipulated when generating the predictive model, but are updated during the optimization process.

3.3 Feature Engineering

Prior to building prediction models and optimizations, feature engineering is done using Pandas [23]. Categorical variables are either converted to binary indicators or converted to binary dummies. Continuous distance variables that measure distance from the University and its key market competitors are converted to binary daily and weekly commute variables. We compute an historic 5-year average yield by high school to serve as a proxy of familiarity and/or social network at the University. We convert the major declared on the application into five major group binary indicators: STEM, Fine Arts, Health, Business, and Humanities. The feature engineering results in 95 features from our data sources on which to build our predictions.

Two important features that affect applicants’ decisions to enroll are the promised amounts of merit-based awards and need-based aid. These features are determined by a specific strategy for allocating awards and aid. As indicated in Fig-

ure 1, these two features are updated during an optimization process that traverses through the solution space to find an optimal one. As the features are updated, the classification model relearns to update its estimate of enrollment probabilities.

3.4 Modeling enrollment

The data is stored in a matrix D where each row represents an applicant’s profile. The first column of D is the binary enrollment variable, y , where y_i being equal to 1 means applicant i enrolled at the University and y_i being equal to 0 means the applicant did not enroll at the University. All other columns of D are features that a classifier uses to learn y . The goal in modeling enrollment is to assess the viability of using popular classification methods to predict enrollment, and to identify and adopt the best method to assist in the optimization of strategies for allocating merit-based awards and need-based aid.

Classifiers. We investigate the performance of several popular classifiers that can predict enrollment probabilities. These include classification methods based on diverse approaches such as support vector, logistic regression, and k-nearest neighbors. We also consider the ensemble methods of random forest, AdaBoost [29], gradient boosting [13], and a more regularized version of gradient boosting known as extreme gradient boosting [7]. Many of these methods do not strictly predict probabilities of a target variable. Rather, they can provide quantities that can loosely be interpreted as probabilities. For example, in case of ensemble methods, which output quantities based on binary decisions of the base learners, we interpret these quantities as probabilities. We used available implementations of these methods in the scikit-learn [24] and xgboost [7] libraries.

Data scaling. Classification methods such as support vector, logistic regression, and k-nearest neighbors operate on distances between data points. Since distances between different features are not of the same scales, the features need to be standardized first. We employ a popular method of data standardization. For each data point, we subtract from it the mean of the training samples, and then divide that by the standard deviation of the training samples.

Performance metrics. We employ multiple metrics to measure the performance of the classifiers from multiple perspectives. Since the data is imbalanced, a single metric, e.g. accuracy, does not meaningfully reflect different aspects of the performance of a classifier. We consider 3 compound metrics: F-score, balanced accuracy, and AUC (area under the ROC curve), which show different aspects of performance. F-score is a useful metric when we are interested in the ability to predict class-1 samples (applicants who ultimately enroll at the university). It combines two individual measures and is defined as $\frac{2 \cdot \text{precision} \cdot \text{sensitivity}}{(\text{precision} + \text{sensitivity})}$, where precision is the probability that a positive prediction is correctly predicted, and sensitivity is the true positive rate or the probability that class-1 samples are correctly predicted. Balanced accuracy is used when we are equally interested in the ability to predict both class-1 and class-0 samples. It gives equal weights to true positive rates (sensitivity) and true negative rates (specificity). AUC is useful when we are interested in the trade-offs between sensitivity and specificity at various

thresholds as it sums up the area under the curve defined by sensitivity (true positive rate) and 1-specificity (false positive rate).

Comparison to baseline. To gauge the performance of each classification method, we compare its performance to that of two baseline methods. The first baseline is a popular method, ZeroR [11], which always predicts the most frequent label in the training set. The second baseline, Stratified, makes predictions based on the distribution of the labels in the training set.

Cross-validation. To determine the performance of each classifier under a metric, we train the classifier using 90% of the data and test it using the other 10% of the data. We experiment with two popular cross-validation methods of partitioning the data randomly into training and testing sets: (1) repeated random subsampling (up to 50 random splits) and (2) k-fold with k=5 and 10. Although the data partitions are randomly generated, in each iteration, we compare the classifiers using the same random partition.

3.5 Optimization of allocation strategies

Algorithm 1 AutoStart-Optimizer(s_0, C, D, δ, S_T)

```

1:  $s \leftarrow s_0$ ;  $best\_strategy \leftarrow s_0$ 
2: Run Stochastic Hill Climb to find  $\Delta_{max}$ 
3:  $T \leftarrow \frac{\Delta_{max}}{\ln 0.5}$ 
4: while progress is still being made do
5:    $T \leftarrow T \cdot \delta$ 
6:   for  $i$  from 1 to  $S_T$  do
7:      $s\_value \leftarrow C.compute\_value(s)$ 
8:      $t \leftarrow RandomNeighbor(s)$ 
9:      $D.update\_strategy(t)$ 
10:     $C.fit(X_D, y_D)$ 
11:     $\Delta = C.compute\_value(t) - s\_value$ 
12:    if  $\Delta > 0$  or with probability  $e^{\frac{\Delta}{T}}$  then
13:       $s \leftarrow t$ 
14:      if  $s$  is better than  $best\_strategy$  then
15:         $best\_strategy \leftarrow s$ 
16:      end if
17:    else
18:       $D.roll\_back\_strategy(s)$ 
19:       $C.fit(X_D, y_D)$ 
20:    end if
21:  end for
22: end while
23: return  $best\_strategy$ 

```

Local search optimization. Simulated annealing [21] is inspired from a physical annealing process, in which an initially hot temperature lets the local search explore the solution space more freely, allowing the adoption of neighbor solutions that are not as good as the candidate solution. As the search goes on, the temperature cools down and the search is more aggressive in finding better solutions. When the temperature is low enough the search essentially becomes a stochastic hill climb. Starting at a high initial temperature, the annealing is known to be able to escape locally optimal solutions and reach a globally optimal solution eventually.

We adopt a modified version of simulated quenching [18], which is a variant of simulated annealing. It is faster than

simulated annealing because it cools temperatures faster. It is shown that in practice simulated quenching was as good as simulated annealing [18].

While Figure 1 provides a high-level description of each iteration of the optimization, Algorithm 1 provides a more detailed description of how the optimization utilizes simulated quenching to find optimal strategies for allocating merit-based awards and need-based aid. The algorithm takes as input an initial allocation strategy (s_0), which is an estimate from the university's current strategy; a classifier C , which learns from historical or updated features to predict enrollment probabilities; an interface D to dataset, which can update features based on a new strategy or rollbacks previous features based on a previous strategy; δ and S_T , which dictate how fast the temperature schedule is decreased and how long the temperature is kept constant, in each quenching step.

The simulated quenching process aims to improve upon the initial strategy, s_0 , by moving from one neighboring strategy to the next. A better neighboring strategy replaces the current one. Additionally, a worse neighboring strategy can also replace the current one with a certain probability (lines 12-13, Algorithm 1), which is determined by the current temperature T and how much worse the neighbor is. At a higher temperature T , a worse neighboring strategy can replace the current one in hopes of getting to an eventual better optimum.

Auto-start simulated quenching To explore the search space liberally at the start of the search, the initial temperature needs to be sufficiently high. This initial temperature is, however, problem dependent. As such, users generally have to experiment with different values to find an appropriate one. We employ a stochastic hill climber to derive an initial temperature to start the simulated quenching process without requiring users to initiate the search by specifying an initial temperature (lines 2-3 in Algorithm 1). We set the initial temperature T to $\frac{\Delta_{max}}{\ln(0.5)}$, where Δ_{max} is the difference between the worst strategy and the best strategy (local optimum) that the stochastic hill climber finds. At this initial temperature, the largest backward move is accepted with probability 0.5. As T decreases, this probability decreases and a backward move is less likely allowed. When T is small enough, the search effectively turns into a stochastic hill climber.

Solution representation. The local search algorithm navigates through the search space of allocation strategies to find an optimal solution (strategy). An allocation strategy consists of a merit-based award allocation strategy and a need-based aid allocation strategy. Each consists of (1) a list of buckets into which students are placed and (2) how much money students in each bucket will receive. Buckets and amounts must be increasing. This means a higher achievement results in higher merit-based amounts, and higher need results in higher need-based amounts. As an example, consider this allocation strategy:

- Achievement buckets: [0, 0.25, 0.50, 0.75, 1]
- Merit amounts: [\$0, \$2000, \$6000, \$10000]

- Need buckets: [\$0, \$500, \$5000, \$10000, \$20000, \$40000]
- Need amounts: [\$0, \$1000, \$1500, \$2500, \$3000]

If an applicant’s achievement index (or need index) falls between bucket $[j]$ and bucket $[j+1]$, then the applicant receives amounts $[j]$. For example, if applicant i has a merit index of 0.4 and a need index of \$15000, then the applicant will receive a merit-based amount of $M_i = \$2000$ and a need-based amount of $N_i = \$2500$. Collectively, the offered amounts of merit-based awards and need-based aid are captured in the features M and N of D .

Generating neighboring strategies. Local search requires the ability to generate a neighboring strategy from a current candidate strategy. The two main steps of how the function RandomNeighbor (line 8, Algorithm 1) generates a neighboring strategy for a given strategy s are as follows:

- 1: First, select with probability 0.5 either the *merit amounts* list or the *need amounts* list of s . Call this list $A = [a_1, \dots, a_k]$.
- 2: Second, with probability 0.5, either add a small amount to a random amount a_r ($1 \leq r \leq k$) in A , or remove a small amount from a random amount a_r in A .

RandomNeighbor keeps repeating these steps of adding or removing small amounts from A until it finds a neighboring strategy that satisfies the following constraints: (1) A remains in an increasing order ; (2) the resulting amount, a_r , must be sufficiently different from its adjacent entries (a_{r-1} and a_{r+1}); and (3) the minimum amount in A cannot be too small (in case of a remove) or the maximum amount in A cannot be too big (in case of an add).

Updating allocation strategies and relearning. To determine if a neighboring strategy t is better or worse than the current one two steps must be taken. First, we have to use this strategy t to update the merit-based awards (feature M of the dataset D) and need-based aid (feature N of the dataset D) for all applicants (rows of D). Second, we have to refit the classifier C to this updated dataset. These two steps are shown in lines 9-10 of Algorithm 1.

If t ends up replacing s , the search moves on to the next iteration. If t does not replace s , we have to roll back to strategy s and refit the classifier C to the previous dataset with strategy s (lines 18-19, Algorithm 1).

Expected outcomes as a result of an allocation strategy. An allocation of funds to awards and aid affects enrollment, which affects multiple outcomes that the University is interested in. We utilized a classification method, e.g. gradient boosting, to predict enrollment probabilities of applicants. This enables us to compute the expected outcomes that we are interested in. Let $y = (y_1, \dots, y_n)$ be the binary target variable enrollment, and let $p = (p_1, \dots, p_n)$, where p_i is the probability that applicant i enrolls at the University.

The expected outcomes we are interested in include:

- Expected enrollment = $\sum_i p_i$. Denote the expected enrollment as E .

- Expected net revenue = $\sum_i p_i \cdot R_i$, where R_i is net revenue obtained from applicant i . $R_i = t_i - (m_i + n_i)$, where t_i is the tuition that the applicant pays, and m_i and n_i are the merit-based and need-based amounts offered to this applicant.

- Expected unmet need = $\frac{1}{E} \cdot \sum_i p_i \cdot (COA_i - EFC_i - Pell_i - M_i - N_i)$, where COA, EFC, and Pell are cost of attendance, expected family contribution, federal Pell grant; and M and N are the University’s merit-based award and need-based aid. A positive unmet need amount is what the applicant is expected to borrow to pay for attending the University. This definition is widely used as a measure affordability.

- Expected accessibility = $\sum_i p_i \cdot I_i$, where $I_i = 1$ iff applicant i is offered some financial aid (i.e. $M_i + N_i > 0$) and has unmet need that exceeds a certain threshold. Note that M_i and N_i are the amounts of merit-based award and need-based aid that applicant i is promised. This threshold is set by the University and is believed to represent a level of need that is not surmountable through existing State and Federal programs.

- Expected return on investment = $\frac{\sum_i p_i \cdot R_i}{\sum_i p_i \cdot (M_i + N_i)}$. It is the expected revenue divided by the expected total promised amounts of awards and aid.

- Expected achievement = $\frac{1}{E} \cdot \sum_i p_i \cdot A_i$, where A_i is the achievement index of applicant i and is calculated from a combination of the applicant’s standardized test scores (e.g. SAT or ACT) and high school GPA. Applicant i is offered a merit-based award in an amount of M_i based on A_i .

Multi-objective optimization. Algorithm 1 aims to find a strategy s^* that maximizes the weighted sum of expected enrollment (enr), net revenue (rev), unmet need (un), accessibility (acc), return of investment (roi), and achievement (ach) as follows:

$$f(s, D, C) = \alpha_1 \cdot E[\text{enr}] + \alpha_2 \cdot E[\text{rev}] - \alpha_3 \cdot E[\text{un}] + \alpha_4 \cdot E[\text{acc}] + \alpha_5 \cdot E[\text{roi}] + \alpha_6 \cdot E[\text{ach}] \quad (1)$$

where α_i ’s are the weights of the expected outcomes. Implicitly, the allocation strategy s is applied to the data D , from which the classifier C learns to predict enrollment the probability p , which is used to compute the expected values on the right hand side of the equation.

Although we do not expect the algorithm to find a strategy that is optimal in each individual objective, an overall optimal value of the function should benefit both the University and the applicants. While higher expected values of enrollment, revenue, and achievement benefit the University, less unmet need and high accessibility benefit applicants. On the one hand, it seems that reducing unmet need for applicants may reduce revenue. On the other hand, making attendance more affordable may actually increase both the expected enrollment, which in turn may increase the expected revenue. In other words, increasing merit-based and need-based amounts may reduce the revenue from each enrolled applicant, but may increase the number of enrolled applicants and, ultimately, the overall revenue.

Table 1: Performance of predicting enrollment

	balanced accuracy	F-score	AUC
Gradient Boosting	0.91	0.88	0.96
XGB	0.91	0.88	0.96
AdaBoost	0.88	0.84	0.95
Random Forest	0.77	0.69	0.92
Linear SVC	0.72	0.61	0.83
KNN	0.61	0.44	0.69
Stratified baseline	0.50	0.30	0.50
ZeroR baseline	0.50	-	0.50

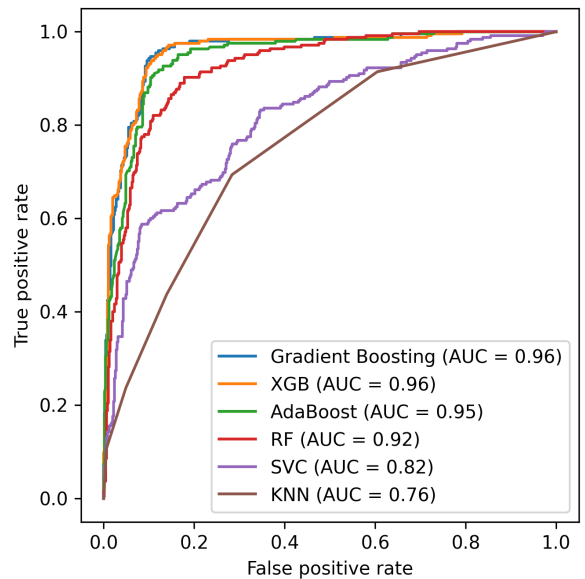
Users can experiment with α 's to give different weights to different objectives to obtain realistically acceptable trade-offs and improvements among the individual objectives.

Constraints. In constructing this research, certain constraints were determined by the University’s administration. These constraints reflect the administration’s perspective on the competitive environment and internal business processes that will support any proposed strategy. These constraints depend on the award structure under consideration. For merit-based awards, there could be no more than 6 awards with award amounts restricted to a specific range. The need-based aid strategy specification was limited to 4 awards with their own maximum and minimum constraints on aid amounts. It is worth noting that the maximum value for a merit-based award was four times higher than the maximum need-based aid amount. Need-based aid eligibility also had a minimum need index cutoff ($COA_i - EFC_i - M_i - Pell_i$). For both allocation strategies there needed to be a minimum difference of \$100 between award buckets.

4. RESULTS

Predicting enrollment. We found gradient boosting [29] was the highest performing classifier and had very high performance in predicting enrollment. This result was obtained by comparing optimized versions of 6 different popular classification approaches, which could make probabilistic predictions. Figure 2 shows the ROC curve of the classifiers in one random partition the data into 90% training and 10% testing sets. The figure shows an excellent trade-off between true positive rate and false positive rate for the two top performers (gradient boosting and extreme gradient boosting).

Table 1 shows the performance of the classifiers averaged across 10 folds of cross validation. In validating the classifiers, we experimented with k-fold cross validation and repeated subsampling at various parameters. We ultimately decided that a 10-fold cross validation was a slightly better choice for our study than the others. All classifiers performed significantly better than the two baselines. Note that given the imbalance of the data, ZeroR did not predict any positive label, resulting in undefined precision and F-score. Gradient Boosting [29] and Extreme Gradient Boosting [7] had the same highest performance across all 3 metrics (balanced accuracy, F-score, and AUC). Gradient Boosting, AdaBoost, and Random Forest were optimized after consid-

**Figure 2: ROC curve of enrollment prediction**

ering various values of maximum depth and minimum leaf size of the decision-tree base learners.

Given an allocation strategy for awards and aid, we employed a gradient boosting classifier to predict expected outcomes if we were to apply the strategy. To assess this strategy, we compare the expected outcomes to the actual outcomes that were obtained from the original data based on the University’s existing strategy for allocating awards and aid.

Defining strategies for allocating funds. A *strategy* for simulated quenching consists of 3 parameters: (1) the number of steps in decreasing an initial temperature, (2) the number of times to keep the current temperature constant, and (3) the weights (α_i 's) of the expected outcomes in Equation 1. For example, we denote **SQ-enr-100,3** as the strategy that optimizes enrollment where temperatures are decreased 100 steps, and each temperature is kept constant for 3 steps. As another example, we denote **SQ-enr-un-2,0.6,100,3** as the strategy that optimizes enrollment and unmet need with weights 2 and 0.6, respectively, where temperatures are decreased 100 steps, and each temperature is kept constant for 3 steps.

Table 2 shows the expected outcomes (averaged across 10 random runs) relative to the baseline (actual outcomes) of a few distinguishing strategies. These expected outcomes are from (1) a strategy for optimizing enrollment; (2) two strategies for optimizing enrollment and unmet need; and (3) seven strategies for optimizing enrollment, unmet need, and revenue.

For clarity, expected outcomes are grouped into two categories: institution-centric outcomes and student-centric outcomes. In general, it is difficult to find a strategy that can improve all of the outcomes simultaneously.

Table 2: Expected outcomes of different strategies versus actual outcomes

Strategy	Institution-Centric				Student-Centric		Budget Change
	Enrollment (%)	Revenue (%)	ROI (%)	Achievement (%)	Accessibility (%)	Unmet need (\$)	Total Funding (%)
SQ-enr-100,3	32.0	29.0	-1.0	-2.0	189.0	3320.0	48.0
SQ-enr-un-2,0.6,100,3	17.0	-26.0	-3.0	-1.0	132.0	-403.0	191.0
SQ-enr-un-2,0.6,100,5	22.0	-25.0	-3.0	-2.0	142.0	-51.0	216.0
SQ-enr-un-rev-1,0.5,2,100,1	5.8	14.6	2.7	-0.4	119.8	1247.0	-28.1
SQ-enr-un-rev-1,0.5,2,100,5	8.1	16.6	2.9	-0.4	125.7	1429.7	-24.5
SQ-enr-un-rev-1,0.6,2,100,1	11.8	6.5	-0.8	-0.7	132.4	948.7	34.9
SQ-enr-un-rev-1,0.6,2,100,2	8.3	6.9	-0.2	-0.5	124.0	825.0	15.5
SQ-enr-un-rev-1,0.6,2,100,3	5.7	1.7	-0.6	-0.4	115.8	345.7	23.2
SQ-enr-un-rev-1,0.6,2,100,4	3.5	-0.2	-0.5	-0.3	109.8	135.2	19.5
SQ-enr-un-rev-1,0.6,2,100,5	2.0	-4.8	-0.4	-0.2	104.3	-288.6	29.9

Table 3: A comparison between the actual outcomes and expected outcomes resulting from a conservative solution for distributing awards and aid

Institution-Centric Outcomes				Student-Centric Outcomes		Budget Changes		
Enrollment	Revenue	ROI	Merit	Accessibility	Unmet need	Total Funding	Awards	Aid
1.3%	0.1%	-0.2%	-0.1%	104.7%	-\$2.7	6.5%	-2.2%	93.4%

Low impact on expected ROI and achievement in optimization strategies. In our evaluation of different strategies, it seems that the impact on expected ROI and merit profiles is relatively small. Percent changes to ROI are reported in column 5 of Table 2. ROI deviates 3% from the baseline in either direction at most, with the majority of deviations being a decrease of 1% or less. Similarly, expected achievement decreases from the baseline by 2% at most, which is equivalent to less than a one point drop in average ACT scores. As such these particular outcomes are not discussed at length in results discussed subsequently.

Optimizing for enrollment alone is good for the University but bad for students. Previous literature focused on optimizing for enrollment alone [4, 28]. In Table 2 this strategy is represented by **SQ-enr-100,3**. Enrollment and net revenue increase 32% and 29% over actual figures, respectively. These results appear attractive if considered in isolation. However, this return comes at the expense of a 48% increase in the total financial aid budget as well as a \$3320 increase in the average amount of unmet need carried by matriculates, indicating attending the University has been made less affordable on average. This increase in expected average unmet need is particularly interesting when compared with the 189% increase in accessibility. Together affordability and accessibility are student-centric outcomes often paired in institutional mission statements. However, there appears to be a trade-off between the two. The extension of more strategic awards and aid to those students with fewer means, may increase their likelihood of attending but it will also raise the average amount of unmet need as the awards and aid are not enough to offset the total cost.

Optimizing for both enrollment and unmet need is good for students but bad for the University. Since optimizing for enrollment alone increases unmet need, we next examine the impact of optimizing for both simultaneously. **SQ-enr-un-2,0.6,100,3** and **SQ-enr-un-2,0.6,100,5** in Table 2 are two such strategies for this dual optimization specification. In both strategies the University sees improvements in terms of enrollment (17 - 22% increase), accessibility (increase 132-142%), and unmet need (decrease \$51 - \$403). However, these improvements are costly the University. To achieve this, total financial aid funding is anticipated to rise 191-216%, causing a reduction in net tuition revenue of 25%. This trade-off between affordability and net revenue necessitates a more nuanced optimization specification.

Optimizing for enrollment, unmet need and revenue results in reasonable trade-offs. If the University wishes to strike a balance between institution- and student-centric outcomes, enrollment, unmet need, and revenue must be simultaneously optimized. Seven different strategies for such a specification are seen in the last two sections of Table 2. From the University’s perspective, each of these strategies results in modest increases in enrollment (2% - 11.8%) and has a negligible to modest impact on expected net revenue (-4.8% - 16.6%). Further, each requires significantly less investment in total funding as compared to optimizing for enrollment or enrollment/affordability and, in some cases, could result in cost savings. From the student perspective there is still a marked gain in accessibility of over 100% in all of the strategies. Unmet need, as a measure of affordability, on average still experiences an increase over the actual amount under all but one strategy, but the amount of the increase in unmet need is always below \$3000, which is a great improvement

Table 4: Effects of allocation strategies on the distribution of merit-based awards and need-based aid

Strategy	Awards (%)	Aid (%)	Total Funding (%)
SQ-enr-un-2,0.6,100,5	160.0	779.0	216.0
SQ-enr-un-2,0.6,100,3	147.0	624.0	191.0
SQ-enr-100,3	54.0	1079.0	48.0
SQ-enr-un-rev-1,0.6,2,100,5	19.6	133.2	29.9
SQ-enr-un-rev-1,0.5,2,100,5	-5.6	441.8	34.9
SQ-enr-un-rev-1,0.6,2,100,3	-0.1	256.4	23.2
SQ-enr-un-rev-1,0.6,2,100,4	-1.2	226.9	19.5
SQ-enr-un-rev-1,0.6,2,100,2	-15.8	328.8	15.5
SQ-enr-un-rev-1,0.6,2,100,1	-57.4	304.9	-24.5
SQ-enr-un-rev-1,0.5,2,100,1	-56.2	253.8	-28.1

over optimizing for enrollment alone.

Few moderate strategic changes can increase accessibility significantly. After we identify a viable strategy, we can narrow it down to a specific solution that maybe adoptable by the University. Adopting a bold strategy that is recommended by an algorithm can be risky. A few reasons that threaten the validity of such a strategy include (1) flawed assumptions made by the model, (2) insufficient amounts of data, and (3) external forces that influence enrollment decisions that are not captured by the data, particularly in a pandemic year. These and other reasons make it hard for administrators to adopt bold strategies even if they might predict large increases in expected outcomes. At times, risk-averse administrators may prefer solutions that make moderate changes, but can move the needle in some significant way.

In addressing this, we identified a viable solution, which came from a random run of the **SQ-enr-un-rev-1,0.6,2,100,4** strategy, which optimized for enrollment, affordability and revenue. Table 3 shows the expected outcomes as a result of adopting this solution.

If this solution is adopted, we expect that accessibility can increase more than 100% from baseline, while keeping enrollment, revenue, and most other relevant outcomes nearly identical as the status quo. For this big expected increase in accessibility to happen, the solution calls for an increase of 6.5% in financial aid funding, and a redistribution of merit-based awards (decreasing 2.2%) to need-based aid (increasing 93.4%).

How allocation strategies affect the redistribution of institution funds. As we assess strategies, we find that different levels of aggressiveness in financial aid spending are demanded. Table 4 illustrates this point by taking each strategy from Table 2 and presenting the percent change in expected spend on merit-based awards, need-based aid, and the total financial aid. As we move down through the table, the level of the impact on total financial aid generally decreases.

Table 5: Performance and runtime from optimizing enrollment of stochastic hill climbing (SHC) and simulated quenching (SQ)

Strategy	Runtime (sec)	Enrollment (%)
SHC-enr-100	8.6	31.4
SQ-enr-100,1	16.9	31.9
SQ-enr-100,2	24.4	32.1
SQ-enr-100,3	34.1	32.2
SQ-enr-100,4	43.8	32.0
SQ-enr-100,5	50.0	32.3

We find that optimizing for both enrollment and unmet need (**SQ-enr-un-2,0.6,100,5** and **SQ-enr-un-2,0.6,100,3**) is a very aggressive approach as the optimization seeks to enroll as many students as possible with the smallest amount of unmet need. This leads to increases in both the expected merit and need spend with an overall budget increase of approximately 200%. A less aggressive strategy is the optimization of enrollment alone (**SQ-enr-100,3**). This strategy seeks enrollment by focusing on expanding need-based aid by 1079% and merit-based awards by only 54%. Since the amount of need-based aid was lower to start with this only results in a 48% increase in the total budget.

It is only when revenue is added to the optimization’s objective with enrollment and unmet need that we reach more moderate budget increases and, eventually, cost savings. These strategies recommend a reduction in merit-based awards and a redistribution of funds to need-based aid.

Comparing simulated quenching and stochastic hill climbing. To contrast the runtime and performance of the two methods, we set both to have 100 iterations. For simulated quenching, temperature is decreased in each iteration, and in each iteration, the temperature is kept constant for another m steps. Thus, in addition to an overhead for estimating the initial temperature, the runtime of simulated quenching should be approximately m times slower than hill climbing. This can be observed in Table 5 for $m = 1, 2, 3, 4, 5$. Similarly, in terms of runtime, simulated quenching is slower than hill climbing by a constant that is directly proportional to m , when both methods were used to optimize for enrollment and affordability (Table 6) and, respectively, for enrollment, affordability, and revenue (Table 7). These tables show the average performance and runtime of the two methods over 10 random runs.

When optimized for enrollment (Table 5), simulated quenching has a higher increase (from 31.9% to 32.3%) than stochastic hill climbing (31.4%). Further, this increase grows with higher values of m (from 1 to 11). When optimized for enrollment and affordability (Table 6), performance also increases with higher values of m (from 1 to 11). However, the increase in performance seems to be on affordability than on enrollment. When optimized for enrollment, affordability, and revenue (Table 7), affordability increases, while enrollment and revenue decrease with higher values of m .

Table 6: Performance and runtime from optimizing enrollment and affordability of stochastic hill climbing (SHC) and simulated quenching (SQ)

Strategy	Runtime (sec)	Enrollment (%)	Unmet Need (\$)
SHC-enr-aff-100	10.2	16.9	-1031.9
SQ-enr-aff-2,0.5,100,1	19.6	26.3	-1193.0
SQ-enr-aff-2,0.5,100,3	39.5	16.7	403.2
SQ-enr-aff-2,0.5,100,5	53.1	21.9	51.4
SQ-enr-aff-2,0.5,100,7	66.4	21.1	205.6
SQ-enr-aff-2,0.5,100,9	84.6	16.1	602.2
SQ-enr-aff-2,0.5,100,11	98.5	18.0	489.6

Table 7: Performance and runtime from optimizing enrollment, affordability and revenue of stochastic hill climbing (SHC) and simulated quenching (SQ)

Strategy	Runtime (sec)	Enrollment (%)	Revenue (%)	Unmet Need (\$)
SHC-enr-aff-rev-100	36.5	11.0	6.4	-1091.8
SQ-enr-aff-rev-1,0.6,2,100,1	19.3	11.8	6.5	-948.7
SQ-enr-aff-rev-1,0.6,2,100,2	121.5	8.3	6.9	-825.0
SQ-enr-aff-rev-1,0.6,2,100,3	160.5	5.7	1.7	-345.7
SQ-enr-aff-rev-1,0.6,2,100,4	197.6	3.5	-0.2	-135.2
SQ-enr-aff-rev-1,0.6,2,100,5	227.5	1.8	-4.5	283.2

In our application, keeping the same temperature longer for simulated quenching (i.e. with large values of m) increases objective scores, but does not seem to produce solutions with more meaningful impacts on expected outcomes.

5. DISCUSSIONS AND CONCLUSIONS

We addressed multiple outcomes for each selected strategy. Two of those outcomes, achievement and ROI, did not see meaningful changes for the strategies selected. In the case of achievement, we could not improve it significantly over the actual outcomes. Perhaps this was due to the fact that as strategies drive the need-based aid budget upward we are successfully recruiting students from lower-income backgrounds who have traditionally scored lower in standardized tests [5]. Or there could be a limit to our ability to recruit high achieving students due to the University-constrained maximum merit-based award amount. For ROI, we could identify strategies that yielded small amounts of increase. However, those strategies were not as compelling as those we chose to include.

In our application, globally optimal solutions may be impractical. For example, a strategy that yields a 200% increase in enrollment might be mathematically optimal, but may not be accommodated easily. One could impose constraints. In our experimentation with the local search approach, however, having constraints on multiple objectives did not lead to great solutions. It can be very hard for a local search to navigate the search space from an initial random solution to get to an optimal solution that satisfies

multiple constraints. Our approach, instead, is to optimize a relatively constraint-free multi-objective function and be optimistic that locally optimal solutions are also practical. Experimenting with different combinations of objectives and ways to control the optimization, we could find strategies that accommodate different levels of risk tolerance.

Thus, this approach provides a path for administrators to weigh risks versus rewards in terms of increasing merit-based awards and need-based aid to support students while maintaining an acceptable level of revenue and enrollment. Table 2 shows the trade-offs in how spending on financial aid funding can affect expected enrollment and revenue. In particular, we were able to identify and recommend a viable solution that requires moderate changes in the budget and yet increases accessibility by more than 100% in Table 3.

Our approach can be adapted to solve other problems that share the same types of interactions between a classifier and an optimizer. In such problems, the classifier expects a set of features that keep evolving based on an external strategy, and the optimizer evaluates expected outcomes predicted by the classifier to recommend a better strategy. For optimization, we employ local search, which is flexible and easily adaptable to different types of problems. For classification, although we employ gradient boosting, any method that is readily available from popular tool kits such as scikit-learn [24] can be used as long as it can make dependable predictions for the problem of interest.

6. REFERENCES

- [1] J. Alhassan and S. Lawal. Using data mining technique for scholarship disbursement. *International Journal of Information and Communication Engineering*, 9(7):1741–1744, 2015.
- [2] C. M. Antons and E. N. Maltz. Expanding the role of institutional research at small private universities: A case study in enrollment management using data mining. *New directions for institutional research*, 2006(131):69–81, 2006.
- [3] A. Assalone, P. DeShawn, and B. McElroy. Unexpected hurdles: Unpacking the price tag of college affordability. Policy brief, Southern Education Foundation, Atlanta, GA, 2018.
- [4] L. Aulck, D. Nambi, and J. West. Increasing enrollment by optimizing scholarship allocations using machine learning and genetic algorithms. *Proceedings of The 13th International Conference on Educational Data Mining*, pages 29–38, 2020.
- [5] L. Barrow and C. E. Rouse. U.s. elementary and secondary schools: Equalizing opportunity or replicating the status quo? In C. Paxson, E. Donahue, C. T. Orleans, and J. A. Grisso, editors, *The Future of children*, volume 16, pages 99–123. Princeton University Press, Princeton, NJ, USA, 2006.
- [6] L. Chang. Applying data mining to predict college admissions yield: A case study. *New Directions for Institutional Research*, 131:53–68, 2006.
- [7] T. Chen and C. Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA, 2016. ACM.
- [8] D. J. Davis, L. C. Green-Derry, and B. Jones. The impact of federal financial aid policy upon higher education access. *Journal of Educational Administration and History*, 45(1):49–57, 2013.
- [9] J. A. Delaney. The role of state policy in promoting college affordability. *The ANNALS of the American Academy of Political and Social Science*, 655(1):56–78, 2014.
- [10] S. L. DesJardins. An analytic strategy to assist institutional recruitment and marketing efforts. *Research in Higher education*, 43(5):531–553, 2002.
- [11] C. L. Devasena, T. Sumathi, V. Gomathi, and M. Hemalatha. Effectiveness evaluation of rule based classifiers for the classification of iris data set. *Bonfring International Journal of Man Machine Interface*, 1:05–09, 2011.
- [12] N. C. for Education Statistics. Total revenue of public degree-granting postsecondary institutions, by source of revenue and level of institution: Selected years, 2007-08 through 2018-19, 2021. data retrieved from Digest of Education Statistics.
- [13] J. H. Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, 29(5):1189–1232, 2001.
- [14] C. F. Goenner and K. Pauls. A predictive model of inquiry to enrollment. *Research in Higher education*, 47(8):935–956, 2006.
- [15] D. Hossler. The role of financial aid in enrollment management. In M. Saddlemire, editor, *New directions for student services*, volume 2000, pages 77–90. Wiley Online Library, Hoboken, NJ, USA, 2000.
- [16] D. Hossler. Enrollment management & the enrollment industry. *College and University*, 85(2):2–9, 2009.
- [17] H. A. Hovey. State spending for higher education in the next decade: The battle to sustain current support. Technical report, National Center for Public Policy and Higher Education, San Jose, CA, 1999.
- [18] L. Ingber. Simulated annealing: Practice versus theory. *Mathematical and computer modelling*, 18(11):29–57, 1993.
- [19] A. W. Johnson and S. H. Jacobson. On the convergence of generalized hill climbing algorithms. *Discrete applied mathematics*, 119(1-2):37–57, 2002.
- [20] A. Juels and M. Wattenberg. Stochastic hillclimbing as a baseline method for evaluating genetic algorithms. *Advances in Neural Information Processing Systems*, 8:430–436, 1995.
- [21] S. Kirkpatrick, C. D. Gelatt Jr, and M. P. Vecchi. Optimization by simulated annealing. *science*, 220(4598):671–680, 1983.
- [22] L. L. Leslie and P. T. Brinkman. Student price response in higher education: The student demand studies. *The Journal of Higher Education*, 58(2):181–204, 1987.
- [23] W. McKinney et al. Data structures for statistical computing in python. In *Proceedings of the 9th Python in Science Conference*, volume 445, pages 51–56. Austin, TX, 2010.
- [24] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.
- [25] L. W. Perna, M. Asha Cooper, and C. Li. Improving educational opportunities for students who work. *Readings on Equal Education*, 22:109–160, 2007.
- [26] L. W. Perna and C. Li. College affordability: Implications for college opportunity. *NASFAA Journal of Student Financial Aid*, 36(1):7–24, 2006.
- [27] A. Rohman, I. Arif, I. Harianti, A. S. Hidayat, W. A. Teniwut, H. Baharun, A. Dja'wa, U. Rahmi, J. R. Saragih, J. Burhanudin, et al. Id3 algorithm approach for giving scholarships. *Journal of Physics: Conference Series*, 1175(1):012116, 2019.
- [28] Z. Sarafraz, H. Sarafraz, M. Sayeh, and J. Nicklow. Student yield maximization using genetic algorithm on a predictive enrollment neural network model. *Procedia Computer Science*, 61:341–348, 2015.
- [29] R. E. Schapire. Explaining adaboost. In B. Schölkopf, Z. Luo, and V. Vovk, editors, *Empirical inference*, pages 37–52. Springer, New York City, NY, USA, 2013.
- [30] R. S. Sexton, R. E. Dorsey, and J. D. Johnson. Optimization of neural networks: A comparative analysis of the genetic algorithm and simulated annealing. *European Journal of Operational Research*, 114(3):589–601, 1999.
- [31] X. Shacklock. *From bricks to clicks: The potential of data and analytics in higher education*. Higher Education Commission London, 2016.
- [32] F. Siraj and M. A. Abdoulha. Uncovering hidden

- information within university's student enrollment data using data mining. In *2009 Third Asia International Conference on Modelling & Simulation*, pages 413–418. IEEE, 2009.
- [33] R. Spaulding and S. Olswang. Maximizing enrollment yield through financial aid packaging policies. *Journal of Student Financial Aid*, 35(1):27–38, 2005.
- [34] P. Sugrue. A model for predicting enrolment yields. *International Journal of Operational Research*, 19(1):60–67, 2014.
- [35] P. K. Sugrue. An optimization model for the allocation of university based merit aid. *Journal of Student Financial Aid*, 40(2):21–30, 2010.
- [36] P. K. Sugrue. A comprehensive merit aid allocation model. *International Journal of Operational Research*, 36:467–476, 2019.
- [37] P. K. Sugrue, A. Mehrotra, and P. M. Orehovec. Financial aid management: an optimisation approach. *International Journal of Operational Research*, 1(3):267–282, 2006.
- [38] D. Trusheim and C. Rylee. Predictive modeling: Linking enrollment and budgeting. *Planning for Higher Education*, 40(1):12–21, 2011.
- [39] S. Walczak. Neural network models for a resource allocation problem. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 28(2):276–284, 1998.
- [40] S. Walczak and T. Sincich. A comparative analysis of regression and neural networks for university admissions. *Information Sciences*, 119(1-2):1–20, 1999.
- [41] D. Whitley. A genetic algorithm tutorial. *Statistics and computing*, 4(2):65–85, 1994.

Automatic Short Math Answer Grading via In-context Meta-learning

Mengxue Zhang
Massachusetts Amherst
mengxuezh@cs.umass.edu

Sami Baral
Worcester Polytechnic Institute
sbaral@wpi.edu

Neil Heffernan
Worcester Polytechnic Institute
nth@wpi.edu

Andrew Lan
Massachusetts Amherst
andrewlan@cs.umass.edu

ABSTRACT

Automatic short answer grading is an important research direction in the exploration of how to use artificial intelligence (AI)-based tools to improve education. Current state-of-the-art approaches use neural language models to create vectorized representations of students responses, followed by classifiers to predict the score. However, these approaches have several key limitations, including i) they use pre-trained language models that are not well-adapted to educational subject domains and/or student-generated text and ii) they almost always train one model per question, ignoring the linkage across question and result in a significant model storage problem due to the size of advanced language models. In this paper, we study the problem of automatic short answer grading for students' responses to math questions and propose a novel framework for this task. First, we use MathBERT, a variant of the popular language model BERT adapted to mathematical content, as our base model and fine-tune it on the downstream task of student response grading. Second, we use an in-context learning approach that provides scoring examples as input to the language model to provide additional context information and promote generalization to previously unseen questions. We evaluate our framework on a real-world dataset of student responses to open-ended math questions and show that our framework (often significantly) outperform existing approaches, especially for new questions that are not seen during training.

Keywords

Automated scoring, Short-answer scoring, Math grading

1. INTRODUCTION

Automated scoring (AS) refers to the problem of automatically scoring student (textual) responses to open-ended questions with multiple correct answers, often utilizing various machine learning algorithms. AS approaches can potentially scale up human grading effort: by training on a

small number of example scores provided by human experts, they can automatically score a large number of responses. With the advancement in online learning platforms in recent years, there has been a growing body of research around the development of AS methods. AS has been studied in many different contexts, including automated essay scoring (AES) [1, 30] and automatic short answer grading (ASAG) [39, 51], which has been studied in various different subject domains [5, 12, 14, 37, 17, 3, 29, 49]. The majority of AS approaches follow two steps: First, obtaining a *representation* of student responses, often using methods in natural language processing, and second, applying a *classifier* on top of this representation to predict the score [4, 27]. Over the years, AS approaches have gradually shifted from classic text representations such as bag-of-words or human-crafted features [8, 16, 19, 30, 32, 41] that are human-interpretable to more abstract representations based on pre-trained neural language models [24, 26, 38, 40, 45].

In this paper, we focus on ASAG in one particular subject domain: Mathematics. Math questions, or questions that involve mathematical reasoning, are ubiquitous in many science, technology, engineering, and mathematics (STEM) subject domains. Recently, several works [5, 14] have studied ASAG for the responses students provide to math-based open-ended questions that (are often concise) include their reasoning or thinking process about a particular concept. As noted in prior work, a key technical challenge in this domain is that student responses to math-based open-ended questions often are a combination of text (natural language) and mathematical language (symbols, expressions, and equations). However, most existing pre-trained language models such as BERT [13] and GPT [7] are not specifically designed for mathematical language. Therefore, existing approaches for math ASAG that do not address the mathematical language present in student responses [5, 14] may not be able to accurately represent student reasoning processes in their responses. On the other hand, existing methods that focus entirely on mathematical language [22, 42, 50] cannot process natural language contained in open-ended responses.

Another significant limitation of existing AS approaches is that, in most cases, we need to train a separate AS model for each question. In contexts such as AES where questions (essay prompts) may not have high similarities, this approach can often be effective. However, in other contexts where reading comprehension or reasoning is involved,

M. Zhang, S. Baral, N. Heffernan, and A. Lan. Automatic short math answer grading via in-context meta-learning. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 122–132, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853032>

multiple questions may be linked to each other through the background information provided. In the context of math questions, many questions share similar skills or are different parts of a multi-step question. Therefore, training a separate AS model for each question would result in models that can only identify typical patterns in student responses to each individual question but cannot really understand how to differentiate good responses from bad ones. It is likely that these models would not be able to generalize well to previously unseen questions, as noted in [12]. More importantly, training a separate AS model for each question may create a significant problem for model storage and management. This problem is especially significant for state-of-the-art AS approaches that fine-tune pre-trained language models that have millions of parameters.

1.1 Contributions

In this paper, we develop an ASAG framework for students’ open-ended responses to math questions. Building on a grand prize winning solution [15] to the National Assessment of Educational Progress (NAEP) Automated Scoring Challenge¹, our framework is based on fine-tuning a pre-trained BERT language model on actual student responses, with several main innovations:

- First, we use MathBERT [35], a version of the popular BERT language model adapted to mathematical content, as our base model. This model is capable of understanding math symbols and expressions to some extent and help us obtain a better representation of open-ended student responses.
- Second, we leverage in-context learning ideas in NLP research [11, 28] and develop an ASAG approach using on multi-task and meta-learning tools (that are popular machine learning tools to promote model generalizability). Specifically, we fine-tune MathBERT with a carefully designed input format that uses *example responses and scores* as additional input (together with question and response texts) to provide additional context of each question. This input format helps us train a shared AS model across all questions and outperforms the current state-of-arts approach [5].
- Third, we show that meta in-context learning leads to highly generalizable AS models. Our intuition on why our approach is highly effective is that, by explicitly using example responses and scores as input, we reduce the AS task to a *similar response finding* task, which is easier for the model to learn.

We evaluate our ASAG framework on a real-world dataset which contains students’ solution processes to open-ended math questions and grades provided by teachers. Through a series of quantitative experiments, we show that our framework (sometimes significantly) outperforms existing approaches in terms of score prediction performance. More importantly, we show that our framework significantly outperforms existing approaches [5, 12] (by up to 50% on some metrics under some settings) when applied to questions that are previously unseen during training, using only a few

¹<https://github.com/NAEP-AS-Challenge/info>

scored examples for these new questions. Perhaps surprisingly, we found that MathBERT does not provide additional benefit on top of the original BERT model while the in-context fine-tuning setup is key to the excellent generalization performance. We also summarize observations from qualitative evaluations of scoring errors, discuss the limitations of our framework, and outline several avenues for future work. Our implementation is publicly available.²

2. RELATED WORK

In recent years, there have been many developments in ASAG methods across various domains. Most of the prior works have focused on non-mathematical domains [6, 10] where student responses are purely textual. However, more AS works have started to focus on more specific domains that contain non-textual symbols, e.g., Math, Physics, Chemistry, Biology and Computer Science [19, 30, 41]. In these domains, a combination of natural language processing methods for the representation of responses and machine learning methods for score classification has shown promising results [5, 14, 23, 36, 38].

Here, we discuss two recent AS works in the mathematical domain, [5] and [12], that are the most relevant to our research. The authors of [5] proposed a scoring approach for short-answer math questions using sentence-BERT (SBERT)-based representation of student responses. Compared to this approach, our approach differs in many aspects and we highlight the following: First, we use MathBERT, a model pre-trained on mathematical content to represent student responses, while the approach in [5] ignores mathematical language in student responses. Second, we use an in-context meta-training approach to train one AS model for all questions while the approach in [5] trains one AS model for each question, which likely limits its generalizability to previously unseen questions.

The authors of [12] proposed a similar scoring approach for short-answer critical reasoning questions that combines various pre-trained representations, including SBERT, with classifiers for AS. Instead of using only student responses as input to the classifiers, they also use a series of question context information such as question text, rubric text, and question cluster identifier. As a result, they showed that their AS approach can generalize to previously unseen questions. Compared to this approach, our approach mostly differs in two aspects: First, we fine-tune MathBERT on actual student responses while the approach in [12] leaves the pre-trained representations fixed, which likely limits the accuracy of their student response representations. Second, we use scoring examples as input to MathBERT in addition to question text to further provide the AS model context of the question, which further enhances the generalizability of our model to previously unseen questions in a *few-shot* learning setting.

3. METHODOLOGY

In this section, we detail both the ASAG setup for math questions and our in-context meta-learning framework.

3.1 Problem Statement

²https://github.com/kikumaru818/meta_math_scoring

We treat math ASAG as a classification problem where our goal is to train a scoring model that is capable of generalizing to new, previously unseen questions using a few examples. This setting is well studied in machine learning, commonly referred to as few-shot learning [7, 11, 28], where the goal is to train robust models that excel at multiple tasks. Formally, we have a set of questions $T = \{Q_1, Q_2, Q_3, \dots, Q_n\}$, where each question $Q_i \in T$ can be seen as a classification task. Each question Q_i comes with numerous graded, training examples: $\{e_i^1, e_i^2, \dots\}$. Each example consists of multiple fields of information: $e_j^i = \langle q_{text}, q_{id}, x, y \rangle$, where q_{text} is the textual statement of the question, q_{id} is a unique question id, x is the text of student’s response and y is the grade from the teacher. We study on two problem settings in this work: i) generalization to new responses and ii) generalization to new questions.

3.1.1 Generalization to new responses

This problem setting follows from that used in prior work [5]: we train a scoring method on scored responses for all questions and test it on held-out responses. We treat this problem setting as supervised learning classification and learn a scoring model $f : x \mapsto \hat{y}$ that predicts an estimated score \hat{y} for a student response x with true score y by minimizing a loss function $L(y, \hat{y})$. For each question Q_i , we split the corresponding scored responses into two subsets, Q_i^{train} and Q_i^{test} , such that $Q_i^{train} \cup Q_i^{test} = Q_i$ and $Q_i^{train} \cap Q_i^{test} = \emptyset$. Instead of treating each question separately and train a model for each, we train one unified model on the union of training datasets for all questions, i.e., $\bigcup_{i=1}^{|T|} Q_i^{train}$. We detail the scoring model and our in-context learning setup in Section 3.3.

Let θ represent the model parameters, the optimization objective \mathcal{L}_i for question i is simply the cross entropy, i.e., the negative log-likelihood loss

$$\mathcal{L}_i(\theta) = \sum_{j:(x_j^i, y_j^i) \in Q_i^{train}} [-\log p_\theta(y_j^i | x_j^i, \dots)].$$

We minimize the total objective that spans all questions

$$\mathcal{L}(\theta) = \sum_{i=1}^{|T|} \mathcal{L}_i(\theta)$$

to learn the model parameters θ .

3.1.2 Generalization to new questions

This problem setting can be formulated as a few-shot (or zero-shot) classification problem: we train a scoring model on scored responses for some questions and test its generalization capability to student responses to held-out questions. We first split the set of questions T into T_{train} and T_{test} such that $T_{train} \cup T_{test} = T$ and $T_{train} \cap T_{test} = \emptyset$. We train the scoring model on all scored responses for the training questions $\bigcup_{i=1}^{|T_{train}|} Q_i$. Let γ represent the model parameters for this problem setting, the optimization objectives for each question and across all training questions change to

$$\mathcal{L}_i(\gamma) = \sum_{j:(x_j^i, y_j^i) \in Q_i} [-\log p_\gamma(y_j^i | x_j^i, \dots)]$$

and

$$\mathcal{L}(\gamma) = \sum_{i=1}^{|T_{train}|} \mathcal{L}_i(\gamma),$$

respectively.

At test time, we applied the trained model to new questions $Q_i \in T_{test}$ to see how it can adapt using few (or zero) scored examples for these new questions. We study two cases: i) we do not update the original model with gradient updates, i.e., γ remains unchanged, which we call the **Meta** setting, and ii) we update γ by backpropagating gradients calculated on a few scored responses for new questions, which we call the **Meta-finetune** setting.

3.2 BERT-based classification

We now detail our scoring method based on fine-tuning a pre-trained language model. BERT [13] is a pre-trained language model that produces contextualized representations of text and is also capable of encoding text. We use MathBERT [35], a variant of BERT pre-trained on a large mathematical corpus containing mathematical learning content ranging from pre-kindergarten (pre-k), high school, to college graduate levels. We use MathBERT as our base language model and fine-tune it on our data for downstream ASAG classification.

Figure 1 visualizes our method. The input to BERT is a sequence of tokens, starting with the [CLS] token, a special symbol added in front of every input during training process for BERT based model. Since [CLS] doesn’t have meaning itself and BERT-based models learn contextualized representations of text, we can use the [CLS] embedding as a representation that encodes the entire input. We then feed the [CLS] embedding to a classification layer followed by softmax [18], obtaining the predictive score class probabilities. A key difference between our work and prior works [5, 12] that use BERT is that we also fine-tune the BERT model, i.e., update its parameters and adapt it to ASAG. Prior works only use BERT-based models to extract the representation of student responses; these methods are not likely going to be effective since they cannot adapt to student-generated content. During training, we backpropagate the gradient on the prediction objective to both i) the classification layer, which is learned from scratch, and ii) BERT, which is updated from its pre-trained parameter values.

3.3 In-context Meta-learning

Our key technical insight is that we need to use a well-crafted input format to provide context to the model and help it adapt to the scoring task for each question. Therefore, instead of only inputting the target student response we want to grade, we also include several other features as the input. These features are important to ground the model in the context of each question. For each possible feature, we also add additional textual instructions as input to the model about the semantic meaning of the feature.

Table 1 shows all possible features we include as model input and the corresponding template. Student response denotes the target responses to be scored. Thus, the corresponding textual instruction is “score this answer.” Since student

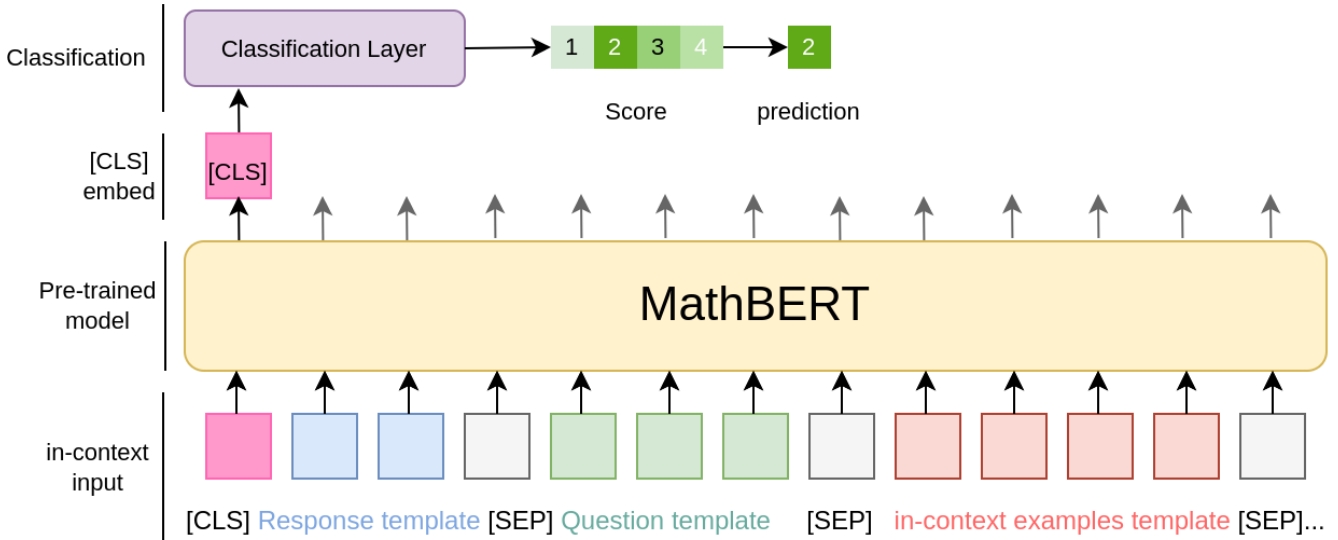


Figure 1: Overview of our in-context meta-learning-based ASAG method for math questions.

Table 1: Templates for different components we use as input into our scoring method.

Input Feature	Template	Sample Text
Student Response	<i>score this answer: x_j^i</i>	<i>score this answer: expand the equation we get $2x + 2 = 1$ then $x = -0.5$</i>
Question	<i>question text: $q_{text_j}^i$</i>	<i>question text: Solve the equation $2(x + 1) = 1$</i>
Question ID	<i>question id: $q_{id_j}^i$</i>	<i>question id: 21314</i>
Scale	<i>scale: possible grade for question i</i>	<i>scale: poor, fair, good, excellent</i>
Example	<i>example: x_{-j}^i, score: y_{-j}^i</i>	<i>example: move 2 to the right $x = 1/2$, score: fair</i>

responses are essential to the grading task, we place it directly after the [CLS] token. After the student response, we add either the question text or the question ID as input to the BERT model. Question text can help the model understand the question context and generalize across questions by leveraging their semantic relations. Question IDs enable the model to identify which question the target response belongs to, which can be helpful when the question text is not semantically meaningful; see Section 4.3 for an example. We can also add textual descriptions of the grading scales to the input. Since we use language models that are better at understanding text than numbers, we use “bad, poor, fair, good, excellent” to represent scores of 0, 1, 2, 3, and 4, respectively.

Another key innovation is that, following recent approaches [11, 28] for meta-training based in-context learning, we also input examples of scored responses, i.e., responses and corresponding scores, (x_{-j}^i, y_{-j}^i) from training dataset that belong to same question of the target response x_j^i . These examples provide further context to the model that the model can use to relate the target response to. Intuitively, when these examples are presented in the input, the AS model only needs to find example responses that are similar to the current response and use their scores to help score the current response. This task is easier for the AS model to learn than the real AS task when examples are not in the input.

4. EXPERIMENTS

This section details the experiments we conducted to validate our in-context meta-training approach for ASAG. Section 4.1 discusses details on the real-world dataset student response dataset we use and how our pre-processing steps. Section 4.2 details evaluation metrics and baselines. We design three groups of experiments to test our approach’s performance. In section 4.3, we examine how the approach performs on generalizing new student responses while having an assumption that the questions have already been seen during the training process. In section 4.4, we examine the performance of our approach generalizing to scoring student response to new questions; in section 4.3.3, we run experiments to test which part of the in-context has the most significant impact on the performance of our approach.

4.1 Dataset

In this study, we use data collected from an online learning platform that has been used in prior work [5, 14]. The dataset contains student responses to open-ended questions paired with scores provided by human graders. The dataset used in [5, 14] consists of 141,612 total student responses from 25,069 students to 2,042 questions, scored by 891 different graders. The numeric score given to each response is in a 5-point scale from 0-4 with 4 as full credit and 0 as no credit. We refer to this dataset as D_{orig} .

D_{orig} contains some noisy data points that increase the difficulty of learning. First, some student responses are the same, but the teacher grades are different. Second, all cor-

responding student responses get full credit for some questions. For example, even the student’s response is “I do not know”, the response’s grade is still full credit. Third, some students’ responses are answered by image, making the text content empty. Fourth, similar issues on question body; some questions do not have semantic meaning (such as questions that refer to a question in a book that we cannot access) or are represented as tables or images. For this work, we mainly focus on questions with corresponding students’ responses and scoring the responses no matter which student is and who is grading. Thus we hope to reduce the effect of these noisy data points and further clean up the dataset. We found that some student responses are the same but the teacher grades are different; therefore, we re-label 2,130 inconsistent responses with the highest grade that the corresponding response text can get. We remove 8,835 student responses that contain only images or broken characters (non-English words, non-math terms). Since our in-context meta-learning approach needs to learn coherent information between questions using question text, we need high quality question text. We remove responses (9,930 number of responses) with a question body (231 number of questions) that does not have semantic meaning. We also remove questions (478) that contain less than 25 number of students’ responses. We called the new dataset D_{clean} , it contains 131,046 responses in total and 1,333 questions. Table 2 shows some examples data points of this dataset. For each data point, it contains the student response, problem text, problem id and teacher grade.

4.2 Metrics

For the evaluation of math ASAG methods, we utilize three evaluation metrics for categorical, integer-valued scores, following prior work [5, 14]. The first metric is area under the receiver operating characteristic curve (AUC), which is designed for binary classification problems. Instead, we calculate the AUC in a way similar to [21] by averaging the AUC numbers over each possible score category, treating them as separate binary classification problems. The second metric is the root mean squared error (RMSE) which simply treats the score categories as numerical values. The third and most important metric is the multi-class Cohen’s Kappa that is often used for ordered categories, which fits the setting of our ASAG data.

4.3 Scoring new responses

4.3.1 Experimental Setting

For this experiment, we focus on comparing the performance of our approach to baselines on generalizing to new responses. We randomly divide all example responses in D_{orig} (we use this dataset for a fair comparison to [5, 14]) into 10 equally-sized folds for cross validation. For each run, we use 8 folds for training, 1 fold for validation to select a training epoch with the best performance on this fold and 1 fold for the final testing of all methods. Under this setting, we ensure every question is contained in the training set so for every response in the test set, our models have seen scored response examples from the exact same question in the training set.

For our approach we use MathBERT [35] as the pre-trained

model with 110M parameters as the base scoring model³. We use the Adam optimizer, a batch size of 16, a learning rate of 1e-5 for 5 epochs on an NVIDIA RTX 8000 GPU. We do not perform any hyper-parameter tuning and simply use the default setting. For each training response, we randomly sample one in-context example per score class and fill up with the rest of training examples up to 25 in total from the training dataset for the corresponding question. Due to the restriction on input length for language models (512 for MathBERT), we truncate an example to a maximum of 70 tokens if necessary to ensure that the question, the target response to score, and all examples all fit in. For testing, we repeat the process of randomly sampling examples eight times for each target student response to be scored and average the predicted score class probabilities.

We use an evaluation setting that follows from the one used in [14], for a fair comparison to compare it with SBERT-Canberra (SBERT-C) [5], the current state-of-the-art method. The evaluation utilizes a 2-parameter Rasch model [44]; We include three groups of terms as covariates in the Rasch model: i) the student ability and question difficulty parameters, ii) the score category predictive probabilities according to the trained scoring method, and iii) the number of words in the response. After training the scoring model, we use the predicted scoring probabilities to learn regression coefficients and the ability/difficulty parameters. Intuitively, this evaluation setup studies how textual information in open-ended responses help *in addition to* student ability and question difficulty during scoring; its purpose is not to evaluate how accurately response scoring models are themselves.

For this evaluation, we use Problem ID as input for each training response to help the model adapt to the task. We do not use question text as input since D_{orig} contains many (709 out of 2,042) question texts that have no semantic meaning (e.g., “For Page 100 question b, answer the question”). This noisy question text cannot help the model recognize different questions and may confuse the model.

4.3.2 Results and Analysis

Table 3 shows the average value for all metrics across the 10 folds for our method (Meta In-context), the SBERT-C baseline, and other baselines studied in [5]. We see that our method is able to achieve a 0.02 (or 4.2%) improvement over the best performing baseline, SBERT-C, on the most important metric, Kappa, while also outperforming on the other two metrics with smaller margins. This improvement validates the effectiveness of our overall method and further pushes the boundary on math ASAG. This improvement is more significant on the cleaned dataset D_{clean} , which we use for further evaluation next. We further note that there is a discrepancy between metric values (high AUC, low Kappa) on this experiment compared to other experiments due to the Rasch model-based setup.

4.3.3 Ablation Study

We conduct an ablation study to verify the effectiveness of each component of our scoring method: using question text as input vs. using only question ID as input, adding textual

³<https://huggingface.co/tbs17/MathBERT>

Table 2: Example questions, student responses, and scores in the dataset.

q_{id} : question unique id	q_{text} : question text	x : student response	y : teacher grade
112348	Write a function rule and a recursive rule for a line that contains the points (-4, 11), (5, -7), and (7, -11)	Don't know what a recursive rule is	0
32147	Ryan had \$800 of his summer job earnings remaining when school started. He plans to use this amount as spending money throughout the 10 months of his school year. please indicate the 3 most important words/phrases in the question	The 3 most important words or phrases in the question are \$800, 10, and months.	4
32149	Ryan will divide the \$800 into 10 equal amounts of \$80. If he completely spends \$80 during each month of his school year, how much of his earnings will remain at the end of the third month of his school year? Explain how you got your answer.	he will have \$560 left. 800-240=560	4

Table 3: Evaluation results using the same dataset and under the same evaluation setting as [14, 5] show that our scoring method outperforms existing methods.

Model	AUC	RMSE	Kappa
Rasch* + Meta In-context (ours)	0.861	0.541	0.496
Rasch* + SBERT-Canberra	0.856	0.577	0.476
Baseline Rasch	0.827	0.709	0.370
Rasch + Number of Words	0.825	0.696	0.382
Rasch* + Random Forest	0.850	0.615	0.430
Rasch* + XGBoost	0.832	0.679	0.390
Rasch* + LSTM	0.841	0.637	0.415

instructions to provide information on the scoring scale, using scored examples to provide additional context, and using MathBERT as the base language model to fine-tune vs. using BERT. For this evaluation, we use the cleaned dataset D_{clean} and a different experimental setting to directly evaluate the scoring accuracy of ASAG methods without using the Rasch model. The rest of the experimental settings, from cross-validation to model training, remain the same. Table 4 shows the results for all variants of our approach on all three metrics. We see that removing question text, textual instructions on scoring scale, and scored examples as input all result in significant degradation in scoring accuracy on some (or all) metrics. Specifically, removing scored examples results in the most significant drop in scoring accuracy, by around 0.02 in Kappa; this result validates the effectiveness that providing in-context examples can significantly benefit language models by helping them adapt to the current task (question). This result clearly validates our intuition that in-context examples reduce the difficulty of the AS task by changing the nature of the task from scoring to finding similar responses, which is easier. Removing question text also results in a (less significant) accuracy drop off: this result directly contradicts our observations in the previous experiment using the original dataset in [5, 14] where we found that inputting the question text results in worse performance than inputting only the question ID. The likely reason for this result is that the cleaned dataset D_{clean} contains much more questions that are semantically meaningful, which are helpful to include in the scoring method to provide important information on the scoring task.

A surprising but important result of this experiment is that using MathBERT results in a small drop off in performance (0.015 on Kappa, 0.007 on RMSE, and a 0.002 improvement on AUC) compared to using BERT. This observation is counter-intuitive since MathBERT is specifically designed to handle math expressions and trained on mathematical content, while BERT is not. To further examine why MathBERT underwhelms on the scoring task, we further investigate its performance on subsets of responses divided according to how much math information is contained in them. Specifically, we divided responses in the test set into two groups according to the amount of mathematical expressions involved: D_{math} that contains responses where more than half of the tokens in the response are mathematical tokens and D_{text} that contains the rest of the responses. Table 5 shows scoring accuracy for our approach using MathBERT and BERT as the base language model on these different response subsets. We see that on responses that are primarily textual, BERT outperforms MathBERT, which suggests that MathBERT loses some ability to encode textual information. On responses that are primarily mathematical, MathBERT performs similarly to BERT on RMSE and Kappa while outperforming BERT on AUC. This result suggests that MathBERT may have some benefit in handling mathematical tokens but the advantage may be minimal. Therefore, an important avenue for future work is to develop language models that are capable of representing and understanding mathematical content.

4.4 Scoring new questions

4.4.1 Experimental Setting

For this experiment, we focus on testing the performance of our approach on generalizing to new questions (tasks) without seeing scored examples and how quickly our approach can adapt to them using few examples. Therefore, we randomly divide all questions in D_{clean} into 5 equally-sized folds in terms of the number of questions instead of the number of responses. As a result, the number of responses in each fold may vary ($26, 229 \pm 689$) since the number of student responses to each question is different. For each run, we use 4 folds for training and 1 fold for testing.

On the test set, we make $n \in \{0, 1, 3, 5, 7, 10, 25, 50, 80\}$

Table 4: Ablation results for different design components of our method on D_{clean} . Most components contribute significantly.

Method Component					Metric		
Question Text	Question ID	Scale	Example	MathBERT	AUC	RMSE	Kappa
✓		✓	✓	✓	0.733 ± 0.006	1.077 ± 0.002	0.589 ± 0.004
	✓	✓	✓	✓	0.724 ± 0.007	1.083 ± 0.003	0.585 ± 0.006
✓		✓		✓	0.710 ± 0.006	1.278 ± 0.002	0.568 ± 0.004
		✓	✓	✓	0.720 ± 0.008	1.088 ± 0.001	0.583 ± 0.009
✓			✓	✓	0.719 ± 0.008	1.091 ± 0.003	0.582 ± 0.005
✓		✓	✓		0.731 ± 0.007	1.051 ± 0.004	0.604 ± 0.010

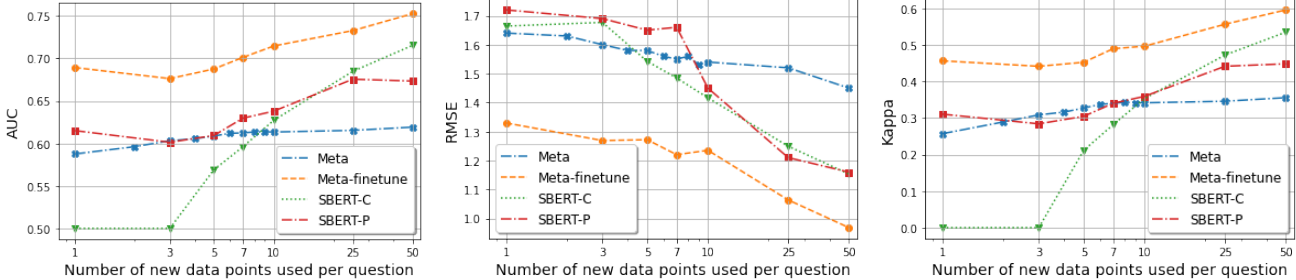


Figure 2: Results on generalizing to previously unseen questions using a few scored examples on all three metrics. Our approach, Meta-finetune, consistently outperforms SBERT-C and SBERT-P. Even without adjusting the model and using the scored examples as input (Meta), we outperform SBERT-C when the number of examples is small.

Table 5: Scoring accuracy on responses that contain more mathematical tokens vs. more text tokens.

Data	approach	AUC	RMSE	Kappa
D_math	MathBERT	0.755 ± 0.008	0.587 ± 0.003	0.690 ± 0.008
	BERT	0.741 ± 0.010	0.610 ± 0.009	0.691 ± 0.020
D_text	MathBERT	0.713 ± 0.006	1.022 ± 0.004	0.523 ± 0.006
	BERT	0.716 ± 0.006	1.001 ± 0.003	0.542 ± 0.008

scored responses per question available to methods trained on the training dataset and evaluate their ability to score other responses. We emphasize that there is no overlap between training responses and test responses for these previously unseen questions. We use two settings for our method. For the first setting, **Meta**, we do not further adjust the trained scoring model; instead, we only feed these responses and their scores, i.e., in-context examples, to the trained scoring model. For cases where $n < 25$, we only feed in n examples even though the method was trained with 25 examples. For cases where $n > 25$, we follow randomly sample 25 examples from the n total examples as input, following the same setting above. This experimental setting can be seen as “*zero-shot*” learning where we directly test how a scoring method trained on other questions works on new questions without observing any scored responses.

For the second setting, **Meta-finetune**, we further fine-tune our trained method on the n new scored responses per question. During this process, for each response as the scoring target, we use the other $n - 1$ responses as in-context examples. This experimental setting can be seen as “*few-shot*” learning where we test how quickly a scoring method trained on other questions can adapt to new questions.

Since **SBERT-C** is the current state-of-the-art math ASAG method on this dataset, we use it as our baseline. According to [5], it calculates similarities between the target response and other responses to the same question. Then it picks the score of the response with the highest similarity to the target response as its prediction, which means that it is not capable of zero-shot generalization to new questions. Therefore, we use n scored examples on these new questions to train the scoring method and evaluate on the other responses. We emphasize again that in both the zero-shot and few-shot settings, the scored examples are excluded from performance valuations.

We also use an additional baseline [12], which we refer to as **SBERT-P**. This method uses SBERT to encode responses and questions and feed the resulting representations to a classifier for predictions. This method also trains a single unified model across and is thus capable of zero-shot generalization to previously unseen questions. We use n scored examples on these new questions for SBERT-P to train on to evaluate it in the few-shot learning setting.

4.4.2 Result and Analysis

Table 6 shows the experimental results averaged over all folds. We see that Meta-finetune outperforms the other three approaches on all values of n for all metrics, achieving satisfactory results of AUC = 0.689, RMSE = 1.329 and Kappa = 0.456 in the one-shot learning setting ($n = 1$), significantly outperforming Meta, SBERT-P and SBERT-C (by up to 50% on Kappa). The performance of Meta-finetune stabilizes as n increases and still outperforms SBERT-C (0.03 on AUC, 0.154 on RMSE and 0.055 on Kappa) and SBERT-P (0.11 on AUC, 0.161 on RMSE and 0.113 on Kappa) at $n = 80$. These results clearly demonstrate that, compared to SBERT-C and SBERT-P, our method is highly

Table 6: Scoring accuracy for different methods on generalization to new questions not seen during training, using a small number of scored examples.

num-of new-data points / question	Method	AUC	RMSE	KAPPA
0	Meta	0.533 ± 0.017	1.650 ± 0.020	0.100 ± 0.052
	SBERT-P	0.558 ± 0.006	1.931 ± 0.001	0.170 ± 0.013
	SBERT-C	—	—	—
1	Meta	0.588 ± 0.012	1.641 ± 0.013	0.257 ± 0.041
	Meta-finetune	0.689 ± 0.033	1.329 ± 0.009	0.456 ± 0.048
	SBERT-P	0.615 ± 0.022	1.721 ± 0.011	0.310 ± 0.043
3	SBERT-C	0.500 ± 0.001	1.664 ± 0.009	0.000 ± 0.001
	Meta	0.606 ± 0.012	1.620 ± 0.013	0.308 ± 0.041
	Meta-finetune	0.676 ± 0.010	1.269 ± 0.010	0.441 ± 0.017
5	SBERT-P	0.601 ± 0.040	1.691 ± 0.010	0.284 ± 0.071
	SBERT-C	0.501 ± 0.001	1.677 ± 0.009	0.000 ± 0.001
	Meta	0.589 ± 0.013	1.581 ± 0.013	0.289 ± 0.043
7	Meta-finetune	0.688 ± 0.009	1.272 ± 0.013	0.452 ± 0.021
	SBERT-P	0.610 ± 0.028	1.650 ± 0.010	0.284 ± 0.050
	SBERT-C	0.569 ± 0.061	1.543 ± 0.080	0.211 ± 0.016
10	Meta	0.611 ± 0.011	1.548 ± 0.011	0.341 ± 0.040
	Meta-finetune	0.701 ± 0.010	1.220 ± 0.008	0.489 ± 0.022
	SBERT-P	0.630 ± 0.037	1.662 ± 0.012	0.340 ± 0.064
80	SBERT-C	0.569 ± 0.006	1.485 ± 0.011	0.282 ± 0.019
	Meta	0.614 ± 0.010	1.543 ± 0.013	0.342 ± 0.043
	Meta-finetune	0.716 ± 0.008	1.235 ± 0.009	0.496 ± 0.021
80	SBERT-P	0.638 ± 0.031	1.453 ± 0.018	0.359 ± 0.080
	SBERT-C	0.627 ± 0.008	1.416 ± 0.009	0.353 ± 0.019
	Meta	0.626 ± 0.024	1.550 ± 0.016	0.373 ± 0.074
80	Meta-finetune	0.765 ± 0.010	0.940 ± 0.015	0.636 ± 0.042
	SBERT-P	0.704 ± 0.033	1.101 ± 0.011	0.523 ± 0.020
	SBERT-C	0.735 ± 0.017	1.094 ± 0.008	0.581 ± 0.042

effective at “warm-starting” scoring models on new questions since it is able to get a sense of how responses should be scored from scored responses to other questions. Again, we note that in-context examples changes the nature of the task from AS to finding similar responses; as a result, models can learn this task quicker and adapt to new questions using only a few examples.

SBERT-C, on the other hand, can barely work in few-shot learning settings, i.e., $n \in \{1, 3\}$. This observation is not surprising since SBERT-C does learn a scoring model from scratch and cannot work when the number of training data points is less than the number of possible score categories. The performance of SBERT-C starts to gradually increase when $n > 5$ but is still significantly worse than Meta-finetune.

Meta, the method for zero-shot learning, although fails to generalize well (only 0.533 in AUC and 0.1 in Kappa at $n = 0$) without seeing any training data, still significantly outperforms SBERT-C with $n \in \{1, 3\}$ and performs similarly to SBERT-P. This advantage only disappears at $n = 10$. To further illustrate this difference, we plot the three metrics vs. n for all methods in Figure 2. We see that Meta’s AUC and Kappa values are higher than that for SBERT-C until n reaches around 8, which indicates that even without re-training the model, it is more suitable for few-shot learning than SBERT-C on new questions.

4.5 Qualitative Error Analysis

In this section, we qualitatively analyze the prediction errors made by our ASAG method. We use the model trained on D_{clean} , with problem text + scale + examples as input into

MathBERT for our analysis.

4.5.1 Feature analysis

To analyze the difference between correct predictions and incorrect predictions, we extract several features that capture properties of the questions and responses to better understand the strengths and weaknesses of the trained ASAG method. As shown in Table 7, “Response math tokens” represents the percentage of math tokens in a response; “Response contains img/table” represents whether a response has images or tables; “Response length” represents the number of tokens a response; “score” represents the actual score given by the graders; “Number of graders” represents the number of graders that graded each response to the question; “question length” represents the number of tokens the corresponding question has and “question math tokens” represents the percentage of math tokens in the question.

Table 7: Features analysis between correct predictions and incorrect predictions. * means the difference is significant ($p_value < 0.005$).

Features (avg.)	Correct Prediction	Incorrect Prediction
Response math tokens (%)*	30.6	25.1
Response contain img/table (%)*	1.29	2.88
Response length*	17.4	29.5
Score*	3.25	2.13
Number of graders	2.53	2.48
Question length*	37.1	39.1
Question math tokens (%)	8.12	7.31

We observe a significant difference ($p_value < 0.005$) between values of the correct predictions and values of the incorrect predictions. We make the following observations:

- The scoring method is more accurate at scoring responses with higher percentage of math tokens and it becomes less accurate when there are higher percentage of plain texts.
- The scoring method is more accurate when the response contains images or tables that words can not represent.
- The scoring method is more accurate at scoring shorter responses.
- The scoring method is more accurate at scoring responses with shorter question description.
- The scoring method is more accurate when the average score of the response is higher. This observation indicates that the model is better at scoring responses with higher quality.
- There is no obvious distinction in grading accuracy for responses with different numbers of graders or to questions with different math tokens percentages.

4.5.2 Question topic and type error analysis

Table 8 lists the summarization of scoring accuracy on different question topics and types. We extract the topics and types from question text using BERTopic [20]. BERTopic is a topic modeling technique that leverages transformers

and term and document frequencies [33] to create easily interpretable topics. Overall, we see that the trained scoring method has better Kappa scores on questions that are primary text-based or involve equations. The result is not surprising since we adapted MathBERT, which likely sees many text-based questions during its pre-training stage. Questions that require students draw graphs in their response also have high Kappa scores; however this result is mainly due to the fact that most of these responses are given full credit, making them easy for scoring methods to make predictions. On the other hand, the trained scoring method has worse Kappa scores on estimation-type and (only a few) multiple-choice questions. This observation can be explained by language models not being trained to capture number sense and thus struggle at numerical reasoning [?]. For multiple-choice questions, the response, i.e., the multiple-choice option, is semantically meaningless, which does not provide meaningful context to the scoring method.

Table 8: Scoring accuracy on different question topics and types. Results are shown in increasing order of the Kappa score. * means the score is better than the average across all responses.

Topic	Type	AUC	RMSE	Kappa
Misc.	Multiple-choice	0.631	1.472	0.400
Math	Table calculation	0.659	1.345	0.445
Algebra	Estimation	0.702	1.310	0.536
Calculus	Estimation	0.716	1.241	0.546
Algebra	Table creation	0.731	0.823*	0.606*
Algebra	Equation writing	0.732	1.023	0.612*
Algebra	Graph drawing	0.734*	0.725*	0.629*
Math	Word question	0.735*	0.663*	0.647*
Calculus	Graph drawing	0.736*	0.610*	0.758*

4.5.3 Error type analysis

For this analysis, we choose a question with scoring accuracy below the average on our dataset to analyze the types of errors made by our trained scoring method. Table 9 shows selected responses with erroneous score predictions and the types of these errors. The question asks students to write an equation with a popular correct response $15/3 = 5$. We make the following observations on typical error types (apart from some obvious human grader errors, which we omit):

- The first error type indicates that our trained scoring method can still struggle on mathematical reasoning and handling numerical tokens. The incorrect responses “ $15*5=3$ ” and “ $5/3=15$ ” have the same numerical tokens but with different ordering and an incorrect operator token compared to the correct response, which completely changes their meaning. The trained scoring method tends to overestimate their scores. This observation suggests that we need base language models with stronger numerical reasoning abilities.
- The second error type indicates that our trained scoring method can struggle with spelling errors in student responses. When the word “equals” is spelled incorrectly in a student response, it does not affect the human grader’s ability to understand the student’s in-

attention. However, the trained scoring method puts a penalty on this spelling error.

- The third error type indicates that our trained scoring method may not recognize paraphrased responses. As shown in the examples, student may add text such as “I think that it is” which does not alter the meaning of the response; however, it adds noise and misled the prediction.
- The fourth error type indicates that our trained scoring method cannot handle responses in unparseable format such as an attachment.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we have proposed a language model fine-tuning-based method for automatic short answer grading for open-ended, short-answer math questions. Our method has two main components: a base MathBERT model pre-trained with educational content on math subjects, and a meta-learning-based, in-context fine-tuning method that promotes generalization to new questions with a carefully designed input format. Experimental results on a large real-world student response dataset revealed surprisingly contradicting findings: Using MathBERT instead of regular BERT, which is not trained on mathematical content, results in a decrease in scoring accuracy, while the in-context fine-tuning method results in significantly improved scoring accuracy compared to existing methods, especially on previously unseen questions.

There are plenty of avenues for future work. First, the observation that MathBERT [35] cannot outperform BERT as the base language model suggests that there is a need to develop more effective models for mathematical language. One promising direction is perhaps taken by another simultaneously proposed version of MathBERT [31] that leverages the inherent tree structure of mathematical expressions. Moreover, the noisiness of human grading that we observed in our experiments suggests that there is a need to develop ASAG methods that take inter-rater agreement into account [43].

Second, there is a need to further improve the completeness of the context information we provide to the base language model. Several possible sources of additional contextual information include the grade level of the question, the common core standard codes, and mathematical skill/concept tags, which can all provide information on the level of the question. Additionally, we may even directly incorporate relevant mathematical content into the model’s input, e.g., by retrieving content chunks in textbooks or online resources using information retrieval methods [9]. However, a potential challenge that needs to be resolved is how to concisely pack all relevant contextual information into the model without exceeding the input length limit of language models (usually 512 tokens).

Third, in order to make ASAG methods more applicable in realworld educational scenarios, there is a need to thoroughly study the fairness aspects of these methods and ensure all students are treated fairly. There is a need to investigate how ASAG methods performs on different student populations; recent work has raised the concern that it is not clear that whether one should explicitly incorporate student

Table 9: Examples of scoring errors made by our trained method.

Question: Chelsea collects butterfly stickers. The picture shows how she placed them. Write a division sentence to show how she equally grouped her stickers. $_ \div _ = _$ Most frequency correct response : $15/3=5$			
Error type	Response	Grade	Predict
Poor reasoning on math operator and numerical token	$15 * 5 = 3$	2	4
	$5/3 = 15$	0	2
	$15_3 = 12$	1	2
Spelling error	3 times 5 eques 15	4	2
Confused by paraphrased responses	I think that it is $5 \times 3 = 15$	4	1
	she place them like in 3 groups and she even did the answer but she did not new the each group	2	0
Meaningless response	see attachment	3	0

demographic information during model training [46]. Future work should explore how to incorporate fairness regularization into the training objective to promote methods that are fair across students [2, 34, 47, 48, 25].

6. ACKNOWLEDGEMENT

The authors would like to thank the National Science Foundation for their support through grant IIS-2118706.

7. REFERENCES

- [1] The hewlett foundation: Automated essay scoring. Online: <https://www.kaggle.com/c/asap-aes>, 2021.
- [2] A. Agarwal, A. Beygelzimer, M. Dudík, J. Langford, and H. Wallach. A reductions approach to fair classification. *arXiv preprint arXiv:1803.02453*, 2018.
- [3] M. Ariely, T. Nazaretsky, and G. Alexandron. Machine learning and hebrew nlp for automated assessment of open-ended questions in biology. *International Journal of Artificial Intelligence in Education*, 1:34, 2022.
- [4] Y. Attali and J. Burstein. Automated essay scoring with e-rater® v. 2. *The Journal of Technology, Learning and Assessment*, 4:3, 2006.
- [5] S. Baral, A. F. Botelho, J. A. Erickson, P. Benachamardi, and N. T. Heffernan. Improving automated scoring of student open responses in mathematics. *International Educational Data Mining Society*, 2021.
- [6] S. Basu, C. Jacobs, and L. Vanderwende. Powergrading: a clustering approach to amplify human effort for short answer grading. *Transactions of the Association for Computational Linguistics*, 1:391–402, 2013.
- [7] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
- [8] J. Burstein. The e-rater® scoring engine: Automated essay scoring with natural language processing. 2003.
- [9] D. Chen, A. Fisch, J. Weston, and A. Bordes. Reading wikipedia to answer open-domain questions. In *Annual Meeting of the Association for Computational Linguistics*, pages 1870–1879, 2017.
- [10] H. Chen and B. He. Automated essay scoring by maximizing human-machine agreement. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1741–1752, 2013.
- [11] Y. Chen, R. Zhong, S. Zha, G. Karypis, and H. He. Meta-learning via language model in-context tuning. *arXiv preprint arXiv:2110.07814*, 2021.
- [12] A. Condor, M. Litster, and Z. Pardos. Automatic short answer grading with sbert on out-of-sample questions. *International Educational Data Mining Society*, 2021.
- [13] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [14] J. A. Erickson, A. F. Botelho, S. McAteer, A. Varatharaj, and N. T. Heffernan. The automated grading of student open responses in mathematics. In *Proceedings of the Tenth International Conference on Learning Analytics Knowledge, LAK '20*, page 615–624, New York, NY, USA, 2020. Association for Computing Machinery.
- [15] N. Fernandez, A. Ghosh, N. Liu, Z. Wang, B. Choffin, R. G. Baraniuk, and A. S. Lan. Automated scoring for reading comprehension via in-context bert tuning. In *International Conference on Artificial Intelligence in Education*, page 0, 2022.
- [16] P. W. Foltz, D. Laham, and T. K. Landauer. The intelligent essay assessor: Applications to educational technology. *Interactive Multimedia Electronic Journal of Computer-Enhanced Learning*, 1(2):939–944, 1999.
- [17] M. Fowler, B. Chen, S. Azad, M. West, and C. Zilles. “Autograding” explain in plain english” questions using nlp. In *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education*, pages 1163–1169, 2021.
- [18] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.
- [19] A. C. Graesser, D. S. McNamara, M. M. Louwerse, and Z. Cai. Coh-matrix: Analysis of text on cohesion and language. *Behavior research methods, instruments, & computers*, 36(2):193–202, 2004.
- [20] M. Grootendorst. Bertopic: Leveraging bert and c-tf-idf to create easily interpretable topics., 2020.
- [21] D. J. Hand and R. J. Till. A simple generalisation of the area under the roc curve for multiple class classification problems. *Machine learning*, 45(2):171–186, 2001.
- [22] A. S. Lan, D. Vats, A. E. Waters, and R. G. Baraniuk.

- Mathematical language processing: Automatic grading and feedback for open response mathematical questions. In *Proceedings of the Second (2015) ACM Conference on Learning@ Scale*, pages 167–176, 2015.
- [23] C. Leacock and M. Chodorow. C-rater: Automated scoring of short-answer questions. *Computers and the Humanities*, 37(4):389–405, 2003.
- [24] S. Lottridge, B. Godek, A. Jafari, and M. Patel. Comparing the robustness of deep learning and classical automated scoring approaches to gaming strategies. Technical report, Cambium Assessment Inc., 2021.
- [25] N. Madnani, A. Loukina, A. Von Davier, J. Burstein, and A. Cahill. Building better open-source tools to support fairness in automated scoring. In *Proceedings of the First ACL Workshop on Ethics in Natural Language Processing*, pages 41–52, 2017.
- [26] E. Mayfield and A. W. Black. Should you fine-tune bert for automated essay scoring? In *15th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 151–162, 2020.
- [27] D. S. McNamara, S. A. Crossley, R. D. Roscoe, L. K. Allen, and J. Dai. A hierarchical classification approach to automated essay scoring. *Assessing Writing*, 23:35–59, 2015.
- [28] S. Min, M. Lewis, L. Zettlemoyer, and H. Hajishirzi. Metaicl: Learning to learn in context. *arXiv preprint arXiv:2110.15943*, 2021.
- [29] R. H. Nehm, M. Ha, and E. Mayfield. Transforming biology assessment with machine learning: automated scoring of written evolutionary explanations. *Journal of Science Education and Technology*, 21(1):183–196, 2012.
- [30] E. B. Page. The imminence of... grading essays by computer. *The Phi Delta Kappan*, 47(5):238–243, 1966.
- [31] S. Peng, K. Yuan, L. Gao, and Z. Tang. Mathbert: A pre-trained model for mathematical formula understanding. *arXiv preprint arXiv:2105.00377*, 2021.
- [32] I. Persing and V. Ng. Modeling prompt adherence in student essays. In *52nd Annual Meeting of the Association for Computational Linguistics*, pages 1534–1543, 2014.
- [33] A. Rajaraman and J. D. Ullman. *Data Mining*, page 1–17. Cambridge University Press, 2011.
- [34] C. Russell, M. J. Kusner, J. Loftus, and R. Silva. When worlds collide: integrating different counterfactual assumptions in fairness. In *Advances in Neural Information Processing Systems*, pages 6414–6423, 2017.
- [35] J. T. Shen, M. Yamashita, E. Prihar, N. Heffernan, X. Wu, B. Graff, and D. Lee. Mathbert: A pre-trained language model for general nlp tasks in mathematics education. *arXiv preprint arXiv:2106.07340*, 2021.
- [36] S. Srikant and V. Aggarwal. A system to grade computer programming skills using machine learning. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1887–1896, 2014.
- [37] C. Sung, T. Dhamecha, S. Saha, T. Ma, V. Reddy, and R. Arora. Pre-training bert on domain resources for short answer grading. In *Conference on Empirical Methods in Natural Language Processing*, pages 6071–6075, 2019.
- [38] K. Taghipour and H. T. Ng. A neural approach to automated essay scoring. In *Empirical methods in natural language processing*, pages 1882–1891, 2016.
- [39] M. Uto and Y. Uchida. Automated short-answer grading using deep neural networks and item response theory. In *International Conference on Artificial Intelligence in Education*, pages 334–339, 2020.
- [40] M. Uto, Y. Xie, and M. Ueno. Neural automated essay scoring incorporating handcrafted features. In *28th Conference on Computational Linguistics*, pages 6077–6088, 2020.
- [41] S. Valenti, F. Neri, and A. Cucchiarelli. An overview of current research on automated essay grading. *Journal of Information Technology Education: Research*, 2(1):319–330, 2003.
- [42] Z. Wang, M. Zhang, R. G. Baraniuk, and A. S. Lan. Scientific formula retrieval via tree embeddings. In *2021 IEEE International Conference on Big Data*, pages 1493–1503, 2021.
- [43] J. Whitehill, T.-f. Wu, J. Bergsma, J. Movellan, and P. Ruvolo. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. *Advances in neural information processing systems*, 22, 2009.
- [44] B. D. Wright. Solving measurement problems with the rasch model. *Journal of educational measurement*, pages 97–116, 1977.
- [45] R. Yang, J. Cao, Z. Wen, Y. Wu, and X. He. Enhancing automated essay scoring performance via fine-tuning pre-trained language models with combination of regression and ranking. *Findings of the Association for Computational Linguistics: EMNLP*, 2020:1560–1569, 2020.
- [46] R. Yu, H. Lee, and R. F. Kizilcec. Should college dropout prediction models include protected attributes? In *8th ACM Conference on Learning@ Scale*, pages 91–100, 2021.
- [47] M. B. Zafar, I. Valera, M. Gomez Rodriguez, and K. P. Gummadi. Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. In *International Conference on World Wide Web*, pages 1171–1180, 2017.
- [48] R. Zemel, Y. Wu, K. Swersky, T. Pitassi, and C. Dwork. Learning fair representations. In *International Conference on Machine Learning*, pages 325–333, 2013.
- [49] X. Zhai, Y. Yin, J. W. Pellegrino, K. C. Haudek, and L. Shi. Applying machine learning in science assessment: a systematic review. *Studies in Science Education*, 56(1):111–151, 2020.
- [50] M. Zhang, Z. Wang, R. Baraniuk, and A. Lan. Math operation embeddings for open-ended solution analysis and feedback. In *Proc. International Conference on Educational Data Mining*, pages 216–227, June 2021.
- [51] Y. Zhang, R. Shah, and M. Chi. Deep learning+ student modeling+ clustering: A recipe for effective automatic short answer grading. In *International Conference on Educational Data Mining*, page 562, 2016.

Investigating Multimodal Predictors of Peer Satisfaction for Collaborative Coding in Middle School

Yingbo Ma, Gloria Ashiya Katuka, Mehmet Celepkolu, Kristy Elizabeth Boyer
University of Florida
{yingbo.ma, gkatuka, mckolu, keboyer}@ufl.edu

ABSTRACT

Collaborative learning is a complex process during which two or more learners exchange opinions, construct shared knowledge, and solve problems together. While engaging in this interactive process, learners' satisfaction toward their partners plays a crucial role in defining the success of the collaboration. If intelligent systems could predict peer satisfaction early during collaboration, they could intervene with adaptive support. However, while extensive studies have associated peer satisfaction with factors such as social presence, communication, and trustworthiness, there is no research on automatically predicting learners' satisfaction toward their partners. To fill this gap, this paper investigates the automatic prediction of peer satisfaction by analyzing 44 middle school learners' interactions during collaborative coding tasks. We extracted three types of features from dialogues: 1) linguistic features indicating semantics; 2) acoustic-prosodic features including energy and pitch; and 3) visual features including eye gaze, head pose, facial behaviors, and body pose. We then trained several regression models to predict the peer satisfaction scores that learners received from their partners. The results revealed that head position and body location were significant indicators of peer satisfaction: lower head and body distances between partners were associated with more positive peer satisfaction. This work is the first to investigate the multimodal prediction of peer satisfaction during collaborative problem solving, and represents a step toward the development of real-time intelligent systems that support collaborative learning.

Keywords

Collaborative Learning, Peer Satisfaction, Pair Programming, Multimodal Learning Analytics

Y. Ma, G. A. Katuka, M. Celepkolu, and K. E. Boyer. Investigating multimodal predictors of peer satisfaction for collaborative coding in middle school. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 133–144, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853010>

1. INTRODUCTION

Collaborative learning benefits learners in numerous ways, such as enhancing critical thinking [31], developing social skills [29], and improving learning gains [32]. During collaborative learning, partners may bring different ideas to solve a problem, defend and evaluate their perspectives, and have a dynamic interaction with each other to produce a shared solution [18]. This relationship between partners can be a decisive factor for the success of the collaboration and positive team experience [9], and the partners' satisfaction toward each other can have a significant impact on their task performance [51] and learning outcomes [20]. Previous literature suggests that students' interactions may not be productive and they may face challenges with their partner [25, 6], which could discourage them from working with partners in the future [44]. In a classroom setting, teachers may not have the resources to detect whether the partners in a team have positive attitudes toward each other and enjoy working together. Therefore, it becomes even more important to develop intelligent and adaptive technologies to predict peer satisfaction during collaborative activities.

Despite the increase in the development of techniques and models to analyze students' interactions during collaborative learning [49, 45], there is no research on automatically predicting peer satisfaction during collaboration. Current studies that analyzed learners' satisfaction during collaboration have revealed important factors such as social presence (sense of being with each other [46, 27]), frequency and quality of team communication [28], and mutual trust between group members [52]). However, most of these post-hoc studies relied on manual approaches (e.g., analyzing post-study attitude survey [22] or open-ended questions [28]). On the other hand, multimodal learning analytics research has created new opportunities to automatically analyze learners' interactions from multiple modalities (e.g., speech, facial expressions, body gestures), and provide insights into the learning process from different dimensions [2]. For example, recent studies successfully classified critical facets of collaborative problem solving process with multimodal features (linguistic, acoustic-prosodic, facial expressions, and task context) derived from groups of learners' collaborative dialogues [48]. However, multimodal learning analytics has not yet been used to automatically predict peer satisfaction from learners' interactions.

Aligned with this motivation, our goal in this paper is to investigate the automatic prediction of peer satisfaction during collaborative learning. We specifically address the following two research questions (RQs):

- RQ 1: What are the most predictive unimodal features of peer satisfaction during collaboration?
- RQ 2: Does multimodal feature fusion improve peer satisfaction prediction compared to the best-performing unimodal model?

To answer these research questions, we analyzed audio and video data collected from 44 middle school learners who worked in pairs on a series of collaborative coding activities. After participating in coding activities, each learner reported their overall satisfaction with their partners. To answer RQ 1, we examined the performance of the following features extracted from learners’ collaborative dialogues, including: 1) linguistic features indicating semantics from Word2Vec [35] and pre-trained BERT [15]; 2) acoustic features such as energy and pitch extracted with openSMILE [17]; 3) eye gaze, head pose, and facial AUs extracted with OpenFace [1]; and 4) body pose extracted with OpenPose [3]. We followed a state-of-the-art methodology [50] that preserves the sequential nature of the features across the collaborative session.

The experimental results revealed two significant predictors. The first significant predictor was head position (x-axis), generated from OpenFace, which was the horizontal distance of a learner’s head from the camera (located in the middle of two learners to collect video recordings). The second significant predictor was body key points (x-axis), generated from OpenPose, which was the the horizontal pixel location of a learner’s eight upper body key points (e.g., nose, neck, and shoulders). These results indicated that learners who had lower head and body distances from their partners were more likely to receive higher peer satisfaction scores. To answer RQ 2, we evaluated the model performance of several early-fused multimodal features, and the results showed that the multimodal features investigated in this study did not significantly improve the prediction accuracy of peer satisfaction compared to the best-performing unimodal feature.

This study provides two main contributions: 1) we present the results from extensive experiments evaluating both a variety of predictive features and a selection of sequential models; 2) and we identify two interpretable and meaningful learner behaviors that can be predictive of peer satisfaction. To the best of our knowledge, this is the first study to investigate the automatic prediction of peer satisfaction with multimodal features extracted from learners’ interactions.

The rest of the paper is organized as follows: Section 2 presents the related work; Section 3 describes the dataset used for this study; Section 4 details the features we investigated;. Section 5 elaborates on the peer satisfaction prediction models; Section 6 presents the experimental settings and results; Section 7 discusses the implications of experimental results; and finally, section 8 concludes the paper and discusses future work.

2. RELATED WORK

Interpersonal interactions and soft skills play an important role in students’ learning experiences and teams’ success during collaboration [11]. Previous research has emphasized that partners may have trouble while collaborating on a task together for a variety of reasons, and many social factors can have an impact on peer satisfaction. For example, So et al. [46] recruited 48 graduate students who collaborated on a healthcare project. They found that learners’ perceived social presence and emotional bonding were important factors for peer satisfaction. Zeitun et al. [51] examined the relationship between team satisfaction and course project performance among 65 groups of students. They found that team satisfaction (toward partners and their collaborative work) were positively related to group performance only for American students, and there was no significant difference in the satisfaction and performance regarding gender. Katuka et al. [26] analyzed the relationship between dialogue act and peer satisfaction from 18 pairs of middle school students. They identified six sequences of dialogue acts (e.g., questions, clarifications) that were positively related to satisfaction. Despite the insights of peer satisfaction provided by the aforementioned studies, most of these studies relied on manual approaches (e.g., post-study attitude survey or open-ended questions), which does not enable the automatic prediction of peer satisfaction.

In recent years, there has been an increasing interest in using multimodal learning analytics (MMLA) techniques that combine multiple data streams (e.g., speech and spoken words [41], text message and facial expressions [14]) to analyze student collaborative interactions. For example, Spikol et al. [47] used MMLA to estimate the success of collaboration with face tracking, hand tracking, and audio recording. They found that distances between learners’ hands and faces were two strong indicators of group performance, and lower distances indicated that it was more likely that successful collaboration occurred among students. Echeverria et al. [16] applied MMLA in a healthcare setting in which nurses collaborated in groups, with their audio, movement, and physiological data collected and analyzed. The authors demonstrated that integrating more sources of data multimodal data provided more contextual details of group activities during collaboration process. In another study, Liu et al. [30] used MMLA to understand learners’ knowledge model refinement process during collaboration. They were able to better predict learners’ knowledge models when they combined multiple data streams (i.e., audio, screen video, webcam video, and log files), which convey important contextual information about student learning. However, to the best of our knowledge, there is no research on automatic prediction of peer satisfaction using multimodal features during collaborative learning. Our study extends this body of MMLA research on learners’ interactions. We investigate different modalities (linguistic, acoustic-prosodic, and visual) for automatically predicting peer satisfaction.

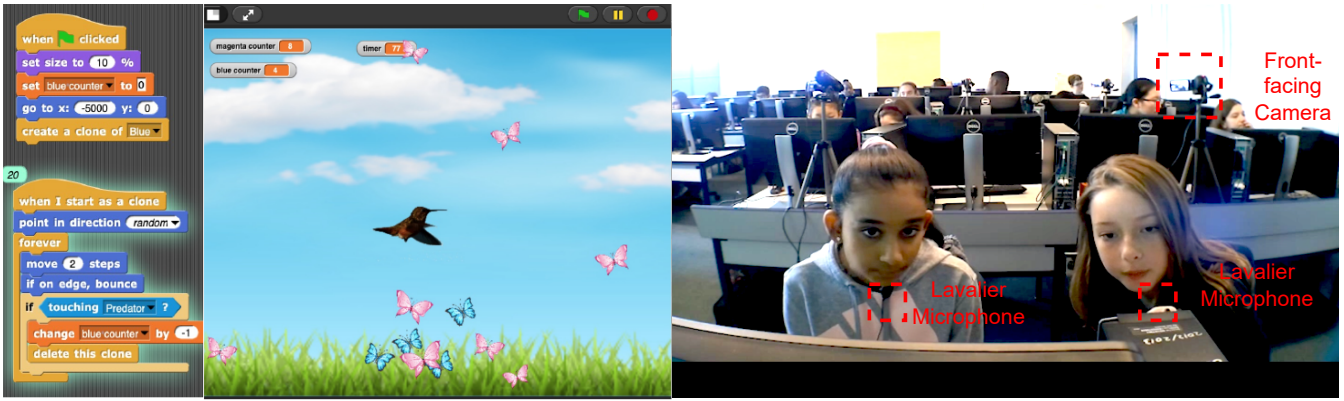


Figure 1: *Left*: A sample script created with Snap!. *Right*: Two middle school learners collaborating on a pair programming task. In the captured moment, the learner in the left side of the frame is the *driver* and the learner on the right is the *navigator*; their collaborative interaction is video-recorded with a front-facing camera and audio-recorded with each learner wearing a lavalier microphone.

3. DATASET

3.1 Participants and Collaborative Activities

Our dataset was collected from 44 learners in 7th grade classrooms in a middle school in the southeastern United States during two semesters (Spring and Fall 2019). Out of 44 learners, 29 (65.9%) identified themselves as females and 15 (34.1%) as males. The distribution of race/ethnicity of these learners included 41.3% self-reporting as White, 26.1% Asian/Pacific Islander, 19.5% Multiracial, 8.7% Hispanic/Latino, 4.3% Black/African American, and 1.9% Other. The mean age was 12.1 with ages ranging from 11 to 13.

The learners collaborated on a series of coding activities in which they practiced fundamental CS concepts such as variables, conditionals, and loops using Snap! block-based programming environment [7]. The learners followed the pair programming paradigm, in which each pair shared one computer and switched roles between the *driver* and the *navigator* during the science-simulation coding activity (Figure 1). The *driver* is responsible for writing the code and implementing the solution, while the *navigator* provides support by catching mistakes and providing feedback on the in-progress solution [4].

3.2 Data Collection and Text Transcription

The collaborative coding session of each pair was recorded at 30 fps in 720p through a front-facing detached camera, and each child wore a lavalier microphone without active noise cancelling. The audio was recorded by digital sound recorders with a sample rate of 48KHz. After the audio/video data collection process was finished, an online manual transcription service [42] generated the textual transcript for each pair. The transcripts included three pieces of information for each spoken utterance: (1) *Starting Time*, in the form of *hour:min:sec*; (2) *Speaker*, in the form of *S1* (the learner sitting on the left of the video) or *S2* (the learner sitting on the right); and (3) *Transcribed Text*. Each collaborative coding session took around 30 minutes. In total, the corpus included 12 hours and 18 minutes of audio and video recordings, with 10,265 transcribed utterances.

3.3 Peer Satisfaction Post Survey

After participating in the collaborative coding sessions, each learner completed a peer satisfaction post survey. To the best of our knowledge, there is no existing validated survey for peer satisfaction in the pair programming context, so we developed a 6-item survey based on previous surveys on peer satisfaction. Sample questions in the peer satisfaction survey included: “My partner answered my questions well”, “My partner listened to my suggestions”, and “My partner often cut my speech”. Each of the six items in the survey was measured on a 5-point Likert scale, ranging from 1 (strongly disagree) to 5 (strongly agree).

Figure 2 (*Left*) shows the distribution of the peer satisfaction post survey responses from 44 learners. The distribution of the satisfaction scores shows that most of the learners agreed or strongly agreed that they were satisfied with the overall interaction with their partner. To determine whether to treat the six post-survey items as a single item or multiple items, we conducted a principal component analysis (PCA). The results of PCA suggested proceeding with only one derived outcome variable, which we refer to as *Satisfaction* score (the average score of six items). This derived outcome explains 52% of the variation across all six survey items, with an eigenvalue of 3.15. Figure 2 (*Right*) shows the distribution of the averaged *Satisfaction* score. The mean value of the *Satisfaction* score is 4.3 ($SD=0.6$) out of 5, with a maximum value of 5.0, and a minimum value of 2.2.

4. FEATURES

In this section, we introduce the feature extraction process from the audio (section 4.1), video (section 4.2), and language (section 4.3) modalities. Then we describe the feature padding process (section 4.4) that prepared the extracted features for model training. Table 1 shows the features this study investigated, and their corresponding dimensional details after the feature padding process.

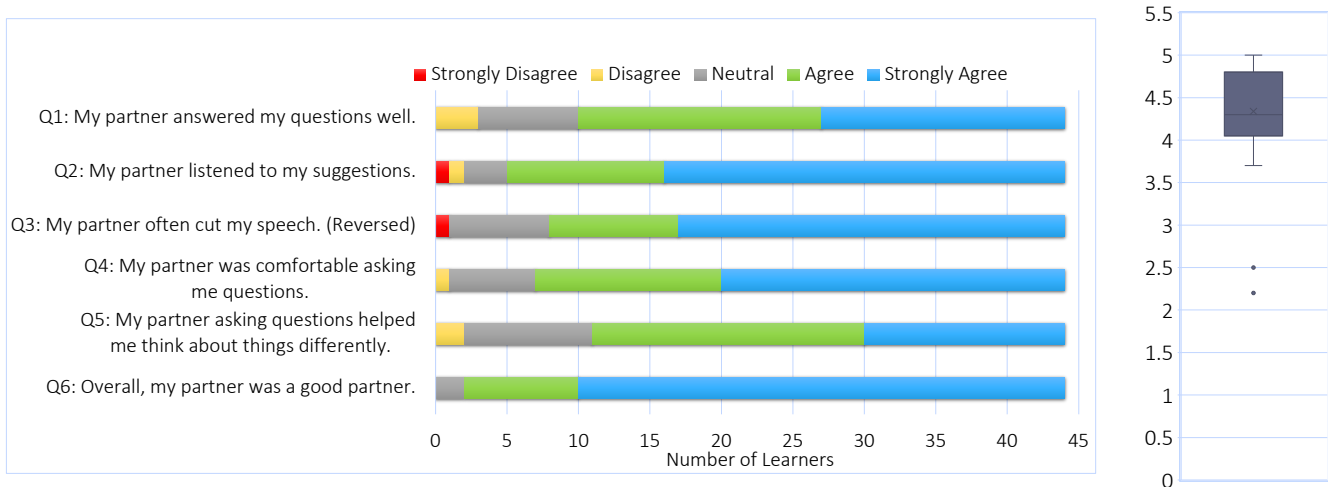


Figure 2: *Left*: distribution of peer satisfaction post-survey items from from 44 learners. *Right*: distribution of the continuous averaged *Satisfaction* score (mean = 4.3, SD = 0.6).

Table 1: Utterance-Level Features

Modality	Feature Name	Vector Dimension*
Audio	Loudness	638, 704, 990
	Pitch	580, 640, 900
	Shimmer	116, 128, 180
	Jitter	116, 128, 180
	MFCCs	928, 1024, 1440
Language	Word Count	1
	Speech Rate	1
	Word2Vec	4,200
	Pre-trained BERT	768
Video	Eye Gaze Directions	784
	Head Directions	294
	Head Position (x-axis)	98
	Head Position (y-axis)	98
	Head Position (z-axis)	98
	Facial AUs	3,430
	Body Key Points (x-axis)	784
Body Key Points (y-axis)	784	

* For every audio-based feature, the three vector dimensions resulted from different speech lengths (29s, 32s, and 45s) after applying three different silence removal thresholds (-6, -16, and -30 dBFS) respectively. For language-derived features, the maximum number of spoken words was 42. For video-derived features, the maximum time length of video segments was 49s.

4.1 Audio-based Features

Simple acoustic-prosodic features (e.g., sound level, synchrony in the rise and fall of the pitch) derived from audio have proven to be effective in predicting learners' engagement level [49] and estimating group performance on solving open-ended tasks [47]. In our study, we extracted audio-derived features on the corresponding audio segment for each utterance. Because we only obtained the *Starting Time* of each utterance from the online transcription service, and not the *Ending Time*, the raw audio segments in our corpus also contain silence (background noise when a

learner stops talking) that elapsed before the next utterance started. To mitigate the potential negative influence of this silence in our audio segments, we used `pydub.detect_silence`, a function in the `pydub` [38] library to detect the time of end-of-utterance in a given audio segment. The function required a pre-defined parameter: silence removal threshold (any audio lengths quieter than this will be considered as silence). For each raw audio segment, we used three different silence thresholds to produce three different audio segments: -6 dBFS (half of the audio's maximum level), -16 dBFS (default setting of the function), and -30 dBFS (low enough to avoid losing actual speech lengths).

After removing the potential silence contained in raw audio segments, we used openSMILE v2.2, an open-source acoustic feature extraction toolkit, for automatic extraction of the following five types of audio-based features within a 20-ms frame and a 10-ms window shift. The five categories of audio-based features are as follows:

1. **Loudness** measures the energy level of the signal. For each audio frame, 11 loudness-related features were extracted.
2. **Pitch** measures the frequency scale of a signal. For each audio frame, 10 pitch-related features were extracted.
3. **Shimmer** measures how quickly the loudness of the signal is changing. For each audio frame, 2 shimmer-related features were extracted.
4. **Jitter** measures how quickly the frequency of the signal is changing. For each audio frame, 2 jitter-related features were extracted.
5. **MFCCs** (Mel-Frequency Cepstral Coefficients) measures the shape of the signal's short-term spectrum. For each audio frame, 16 MFCCs-related features were extracted.

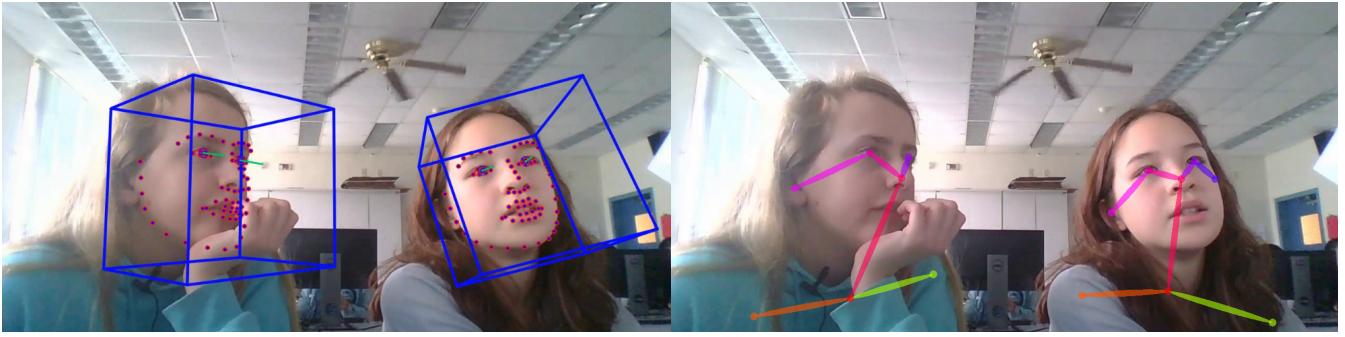


Figure 3: An example of the video-derived feature extraction process for both learners in a specific video frame. *Left*: eye gaze direction (green vectors), head pose (blue 3D bounding boxes), and facial AUs (recognized later) extracted with OpenFace. *Right*: upper body key points (e.g., nose, neck, and shoulders) extracted with OpenPose.

4.2 Language-based Features

Linguistic features extracted from spoken utterances have been used to model collaborative problem solving skills and predict collaborative task performance [37]. In our study, multiple commonly used statistical and semantic linguistic features were extracted for each spoken utterance. The four categories of language-derived features are as follows:

1. **Word Count** For each utterance, word count was calculated as the number of words.
2. **Speech Rate** For each utterance, speech rate was calculated as the number of words divided by the number of elapsed seconds in the utterance, to produce words per second.
3. **Word2Vec** is a semantic method which learns word associations from the text, and groups similar words together in a vector space based on their semantics. We train our Word2Vec model with *gensim*, an open-source natural language processing library. The default settings of parameters were used, in which the dimension of each word embedding was set to 100, with a sliding window size of 5.
4. **Pre-trained BERT** is a language model trained on a large amount of data (e.g., texts from Wikipedia and books) in a self-supervised way. Similar to Word2Vec, BERT represents semantics of words in a vector space. In this study, we used the BERT-base-uncased model, which is a publicly available BERT model trained only on English texts with the hidden size of 768. With this pre-trained BERT, we generated one 768-dimensional vector for each utterance.

4.3 Video-based Features

A variety of features generated from video modality have been investigated in prior literature modeling collaborative problem solving. For example, eye gaze has proven effective in evaluating learners' attentiveness [43, 24] and learning performance [5, 40]; head pose has also been used for assessing learners' collaborative problem solving competence [13]; facial action units (AUs) have been used to measure both individual learners' tutoring outcomes [21] and interaction level during collaborative learning [33]. Body pose has been

used for analyzing learners' engagement level [39] and modeling collaborative problem solving competence [13]. In our study, video-derived features were extracted from the corresponding raw video segment of each utterance. We used the OpenFace v2.0 facial behavior analysis toolkit and OpenPose v1.7 body key points detection toolkit to extract the following four categories of video-based features (See Figure 3):

1. **Eye Gaze Direction** refers to the direction in which an eye looks. For each detected face per video frame, 8 eye gaze direction-related features were extracted. They included 3 eye gaze direction vectors (x direction, y direction, and z direction) for each eye, and 2 eye gaze directions in radians averaged for both eyes.
2. **Head Pose** refers to head position and direction. For each detected face per video frame, 6 head-related features were extracted with OpenFace, including three head position vectors (x direction, y direction, and z direction) representing the location of the head with respect to the camera in millimeters, and three head direction vectors in radius with respect to the camera. Since the front-facing camera was located in the middle of two learners during the data collection process, positive values of the x direction vector and the z direction vector indicate that the learner is sitting on the right side of the video and away from the camera, and vice versa. The head position features used in our study were the absolute values of the x , y , and z direction vectors, representing the spatial location of each learner's head from the camera.
3. **Facial AUs** refer to the movements of an individual's facial muscles. For each detected face per video frame, 35 facial AU-related features were extracted with OpenFace, including 17 facial AU intensity features (ranging from 0 to 5), and 18 facial AU presence features (0-absence or 1-presence).
4. **Body Pose** refers to the location of each joint (e.g., neck, shoulders) of the human body, which are known as key points that can describe a person's pose. For each learner appearing in each video frame, the 2D locations (x direction and y direction) of 8 body key

points, measured in pixels, were extracted with OpenPose. These included the locations of each learner’s eyes, nose, neck, and shoulders. OpenPose supports real-time detection of 25 full body key points (hand, facial, and foot key points); however, since our video recordings only captured learners’ upper bodies, OpenPose was not able to detect the locations of some body points such as hand and foot. Therefore, only 8 body key points related to learners’ upper bodies were extracted and used in this study. Because the resolution of our cameras was 720p, the maximum pixel value of body key points generated from OpenPose was 1280 pixels in the x direction, and 720 pixels in the y direction.

4.4 Feature Padding

Spoken utterances naturally vary in time length, and feature padding is an important step for ensuring the uniform size of model inputs before training machine learning models. We averaged the audio-based and video-based features across a small non-overlapping time window because they were extracted on the frame level. Following the feature aggregation methods used in prior works [47, 49], in which the average time windows of 500 ms and 1000 ms were chosen respectively, we selected the time window of 500 ms. We did not choose a longer window because audio-based features (e.g., pitch) could vary over a longer duration, which would lead to losing fine-grained details. Finally, post padding (adding zeros to the end of vectors) was applied on each averaged feature vector with the maximum time length (29s, 32s, and 45s) for different silence removal thresholds. For the Word2Vec-based feature, word embeddings were concate-

nated to form one feature vector for each utterance. Then, post padding was applied to the Word2Vec-based feature vector and the BERT-based feature vector with the maximum number (42) of spoken words.

5. PREDICTION MODELS

Figure 4 depicts the architecture of our peer satisfaction prediction model. For a collaborative coding session of a given pair (Learner A and Learner B), the model input is a session-level feature sequence $X = [x_0, x_1, \dots, x_{N-1}]$ for Learner B, in which N is the number of spoken utterances from the learner. For each element in X , we used the early fusion method to generate utterance-level multimodal feature $x_t = [a_t, v_t, l_t]$ by concatenating unimodal audio-derived feature a_t , video-derived feature v_t , and language-derived feature l_t . Before concatenating unimodal feature vectors into a single multimodal feature vector, we applied z-score normalization to all the features by subtracting their mean value and dividing by their standard deviation.

Our prediction model contains two stages: feature learning and regression. In the feature learning stage, we followed the current state-of-the-art methodology [50] that preserves the sequential nature of dialogue to learn the input feature sequence X . The sequential model is a two-layer LSTM network with 128 units. We obtained a final 128-dimensional hidden state h_T from the sequential model. During the regression stage of the model, we used h_T as input, and fully connected layers to output a continuous estimated satisfaction score \hat{y} , in order to approximate the actual *Satisfaction* score y rated by Learner A.

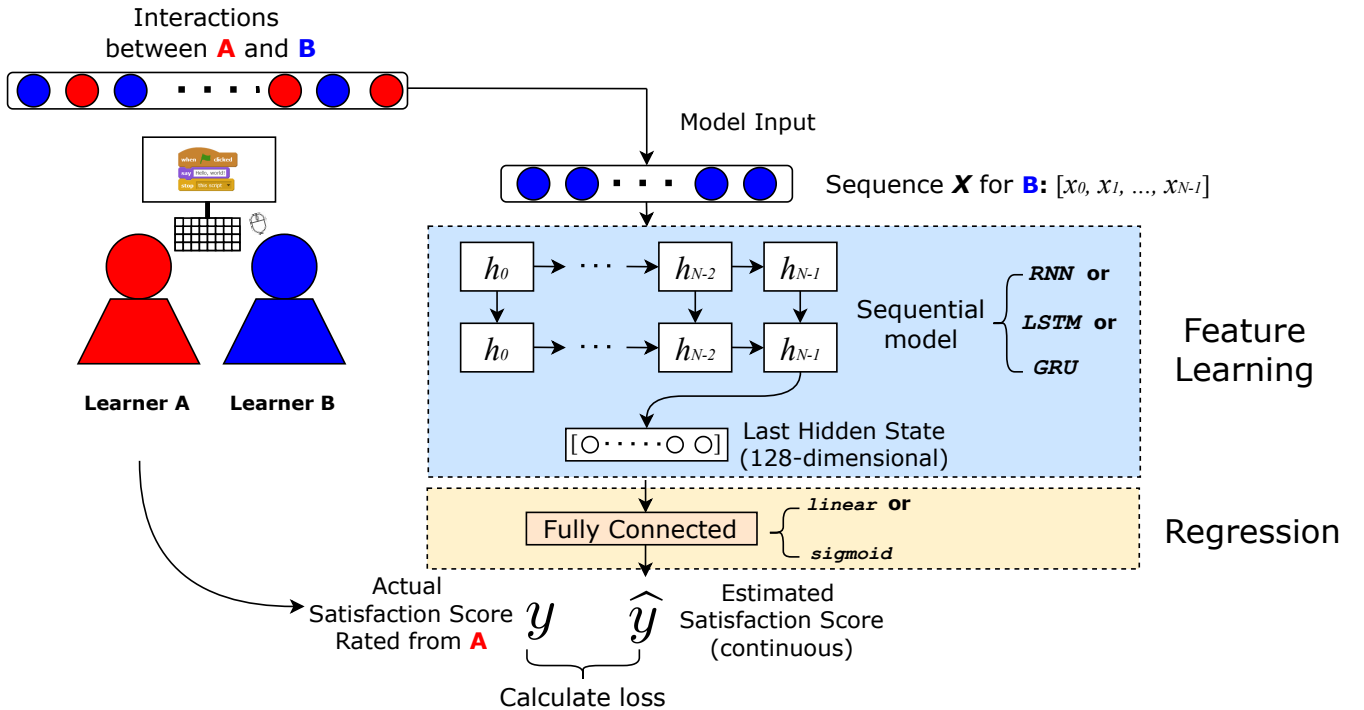


Figure 4: Architecture of the prediction model. For unimodal modeling, x_t ($0 \leq t \leq N - 1$) is a unimodal feature vector (audio a_t , video v_t , or language l_t). For multimodal modeling, x_t is a subset of an early-fused vector $[a_t, v_t, l_t]$ (normalized).

Table 2: Regression results of unimodal models. Two highlighted features: Head Position (x-axis) and Body Key Points (x-axis), significantly reduced the *MAE* compared to the baseline feature (*p-value* < .05).

Modality	Unimodal Feature	<i>MAE</i>	<i>p-value</i> (\hat{y}_{base} and \hat{y})	R^2 (y and \hat{y})
	Baseline	0.1953	—	0.07
Audio	Loudness	0.1981, 0.1790, 0.1796	0.31, 0.19, 0.12	0.02, 0.05, 0.07
	Pitch	0.2073, 0.1902, 0.1881	0.25, 0.14, 0.15	0.01, 0.15, 0.03
	Shimmer	0.1895, 0.1794, 0.1713	0.12, 0.19, 0.42	0.04, 0.12, 0.20
	Jitter	0.1983, 0.1896, 0.1853	0.14, 0.31, 0.31	0.01, 0.08, 0.06
	MFCCs	0.2341, 0.2405, 0.2318	0.19, 0.19, 0.24	0.01, 0.03, 0.03
Language	Word Count	0.1794	0.43	0.07
	Speech Rate	0.1790	0.19	0.29
	Word2Vec	0.1751	0.09	0.06
	Pre-trained BERT	0.1789	0.06	0.08
Video	Eye Gaze Directions	0.1689	0.10	0.21
	Head Directions	0.1583	0.09	0.23
	Head Position (x-axis)	0.1402	0.03	0.68
	Head Position (y-axis)	0.1902	0.21	0.15
	Head Position (z-axis)	0.1640	0.09	0.25
	Facial AUs	0.1927	0.19	0.11
	Body Key Points (x-axis)	0.1376	0.03	0.64
	Body Key Points (y-axis)	0.1761	0.39	0.10

MAE: aggregated testing absolute error for all data samples. y : actual satisfaction scores. \hat{y} : predicted satisfaction scores with each unimodal feature. \hat{y}_{base} : predicted satisfaction scores with the baseline feature. R^2 : another widely used metric to evaluate a regression task’s level of goodness-of-fit.

Table 3: Regression results of multimodal models. None of the multimodal features significantly outperformed the baseline feature.

Multimodal Feature	<i>MAE</i>	<i>p-value</i> (\hat{y}_{base} and \hat{y})	R^2 (y and \hat{y})
Baseline: Body Key Points (x-axis)	0.1376	—	0.64
Head Position (x-axis, z-axis)	0.1484	0.39	0.65
Head Position (x-axis), Head Directions	0.1355	0.17	0.68
Head Position (x-axis), Body Key Points	0.1367	0.10	0.68
Head Position (x-axis), Pre-trained BERT	0.1409	0.13	0.65

Recent research [50] has shown that the type of sequential model can play an important role for feature learning. Therefore, we also evaluated the performance of recurrent neural network (RNN) and gated recurrent unit (GRU) models to understand the influence of different sequential model architectures during feature learning. In addition, we evaluated the performance of two different output units, *sigmoid* and *linear* functions, to compare between linear and non-linear regression.

6. EXPERIMENTS AND RESULTS

6.1 Experimental Setups

We implemented the Python code¹ for our prediction models in Keras with a Tensorflow backend. We conducted five-fold cross-validation to train and validate the models. All labels (y) were normalized (ranging from 0 to 1) before the model training process because the *sigmoid* activation function was used to produce the predicted satisfaction scores \hat{y} . We used Adam optimizer with the learning rate of $1 \times e^{-3}$ to train the prediction model, which was trained for up to 100 epochs. The mean absolute error (*MAE*) was calculated for the loss function. After five rounds of cross-validation, we aggregated the *MAE* of each fold during the model testing process.

¹<https://github.com/yingbo-ma/Predicting-Peer-Satisfaction-EDM2022>

6.2 Investigating Unimodal Features

To identify predictive unimodal features, we compared the prediction accuracy of each unimodal feature with a randomly generated baseline feature. Followed a common method of generating uniform random baselines [12, 19], we used the Python function *random.uniform*(0, 1), which can be interpreted as white noise without any meaningful content. We then trained the model with the white noise to generate the random baseline results (error MAE_{base} and predicted scores \hat{y}_{base}). This low baseline allows us to measure the extent to which each feature predicts the outcome better than random chance. Next, we trained the model with each of the unimodal features from Table 1, and generated corresponding *MAE* and \hat{y} . A paired-samples *t*-test [34] between \hat{y}_{base} and \hat{y} checked whether adding that unimodal feature significantly reduced error compared to the random baseline. Table 2 shows the regression results of peer satisfaction prediction models trained on unimodal features.

For audio-derived features, the three values in each column (from left to right) resulted from different silence removal thresholds (-6, -16, and -30 dBFS). Although time-domain features (e.g., Loudness, Shimmer) performed better than frequency-domain features (Pitch, Jitter), as indicated by lower *MAEs*, the associated *p-values* showed that none of the acoustic and prosodic features significantly outperformed the baseline. For video-derived features, we identified two predictive unimodal features: learners’ head posi-

tions in the x direction (p -value = 0.03), and the locations of their body key points in the x direction (p -value = 0.03). Models trained on language-based features yielded similar $MAEs$ compared to the baseline model; therefore, none of the language-based features evaluated in this study were predictive for this task.

The feature space in our study is large compared to the relatively small corpus size. Therefore, identifying predictive unimodal features helped with filtering out noisy features that are not useful in predicting satisfaction scores. Next, we examined the performance of multimodal models by combining the unimodal features that were useful.

6.3 Examining Multimodal Features

For testing the performance of combining multiple features, we selected the two significant ($p < 0.05$) unimodal features (Head Position x-axis and Body Key Points x-axis). In addition, we also selected Head Direction and Pre-trained BERT, as their p -values are lower than 0.1 (a threshold that has been used to identify a weak trend or association [23]). We used the best-performing unimodal model trained on Body Key Points (x-axis) as the baseline (predicted satisfaction scores \hat{y}_{base}), and investigated the p -values of the paired-samples t -test between the predicted scores \hat{y} and the baseline results \hat{y}_{base} . Table 3 shows the regression results of peer satisfaction trained on unimodal features.

The results shown in column 2 of table 3 indicated that combining Head Position (x-axis) and Head Directions yielded the lowest MAE. However, none of these multimodal features significantly improved the regression performance compared to the unimodal model.

6.4 Comparing Different Model Architectures

To understand the influence of different sequential models during feature learning, and compare the performance between linear and non-linear regression models, we selected the best-performing unimodal model and examined how prediction accuracy varied under different model architectures.

Table 4 shows the experimental results with different model architectures. For the selection of different sequential models, three models provided comparable performances, with LSTM yielding a slightly lower MAE. As for the selection of different activation functions, the model predicting satisfaction score with a *sigmoid* activation function performed better than with a *linear* function. In addition, although we observed faster convergence speed with *linear*, *sigmoid* provided more stable training and testing performance (see Figure 5). As for the selection of the number of layers, the one-layer LSTM performed similarly compared to two- or three-layer LSTM.

7. DISCUSSION

This study investigates the prediction of peer satisfaction using multimodal features from learners’ interactions during collaborative learning activities. This section discusses the results with respect to our two research questions, as well as implications from comparing the performance of different model architectures.

Table 4: $MAEs$ under different architecture settings.

Sequential Model	LSTM	RNN	GRU
<i>MAE</i>	0.1376	0.1401	0.1382
Output Unit	<i>sigmoid</i>	<i>linear</i>	
<i>MAE</i>	0.1376	0.1741	
# of Layers	1	2	3
<i>MAE</i>	0.1359	0.1376	0.1384

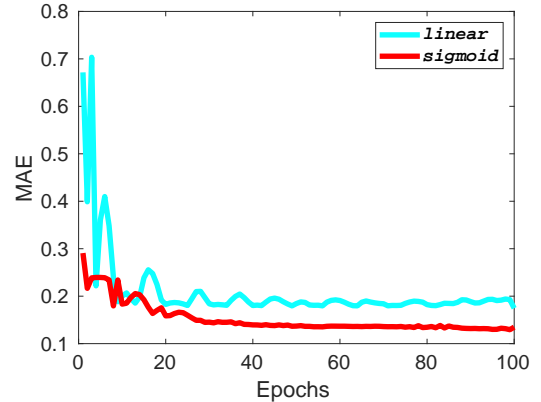


Figure 5: Testing $MAEs$ under different activation functions (blue-linear, red-sigmoid). *linear* provided faster converge speed during training, while *sigmoid* provided lower MAE and more numerical stability during testing.

7.1 RQ 1: What are the most predictive unimodal features of peer satisfaction during collaboration?

7.1.1 Audio-based Features

In this study, we investigated several commonly used acoustic-prosodic features and the results showed that none of these features were significant predictors of peer satisfaction. Previous literature has associated learners’ peer satisfaction with their emotional bonding [46]. Acoustic-prosodic features have been widely used for detecting speaker emotion detection (positive, neutral, and negative) [10] and predicting learners’ task performance [47]. However, the results from this study indicate that the acoustic-prosodic features we tested may not have the explanatory power to predict peer satisfaction.

One potential reason for audio features not performing well in models is if there are periods of silence included in what the model thinks are periods of speech only. We examined several different silence removal thresholds (-6, -16, -30 dBFS) and the results indicated that this strategy did not help with peer satisfaction prediction. A higher silence removal threshold (e.g., -6 dBFS) could help reduce the negative influence from background noise; however, it is also more likely to remove learners’ speech. While selecting between and qualitatively examining different thresholds, we determined -30 dBFS was optimal for our corpus to balance between eliminating periods of silence without excessively cutting off speech. However, acoustic features under that threshold were not predictive of peer satisfaction.

7.1.2 Language-based Features

We examined several statistical (word count and speech rate) and semantic (Word2Vec and BERT) features from the language modality. Statistical features such as word count per utterance and speech rate have shown to be associated with learners’ active participation and turn-taking during collaboration [49]. The results from our study showed that there was a trend toward significance when more semantic information was added to the features (p-values for word count, Word2Vec, and BERT: 0.46, 0.16, 0.06); however, none of these models yielded statistically significant results for predicting peer satisfaction (Table 2). Previous literature also found several sequences of dialogue acts representing speakers’ intentions (e.g., questions followed by clarifications) that were positively related to peer satisfaction [26], but our results did not show a direct correlation between semantics and peer satisfaction. One potential reason may be that the semantic representation methods used in our study did not have the same explanatory power as dialogue acts to directly indicate learners’ intentions.

7.1.3 Video-based Features

Among the several video-based features extracted in this study, head position and body location on the horizontal axis were the only two predictive unimodal features. To better understand how the patterns of these two predictive features varied among learners with different satisfaction scores, we selected three groups of five learners and examined their sessions in more detail. The groups are as follows:

- High satisfaction group: five learners who received the highest scores (5.0 / 5.0).
- Average satisfaction group: five learners who received the exact score of 4.3 / 5.0 (mean peer satisfaction score of our corpus).
- Low satisfaction group: five learners who received the lowest five scores (all below 3.7 / 5.0).

Figure 6 shows the patterns of horizontal (x-axis) head distance from the camera, in meters, for the three groups of learners. For each group, we calculated their averaged Head Distance (x-axis) over whole sessions. Since the camera was positioned horizontally in the middle between two learners, if learners had lower head distance from the camera, this likely reflects that the learners were sitting closer to one another. From Figure 6 we could see that learners who received high satisfaction scores (green) had lower head distances over the collaborative coding sessions, compared to learners who received average (red) and low (blue) satisfaction scores.

Figure 6 also depicts the difference of the head distance variance over time across the three groups of learners. Learners who received high satisfaction scores (green) had lower head distance variance and fewer numbers of sharp distance increases over time, compared to learners who received average (red) and low (blue) satisfaction scores. A sharp head distance increase could happen when the learner became disengaged in the collaborative coding tasks (e.g., talking to learners in other groups). In comparison, for learners in the high satisfaction group (green), only a small range of head distance variance over time was observed.

In addition to head distance (x-axis), another predictive unimodal feature identified in our study was *body key points*,

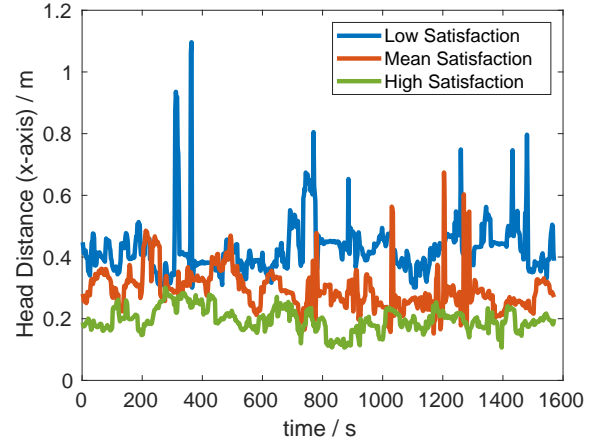


Figure 6: Head Distance (x-axis) in Meters from OpenFace

such as the location of nose, neck, and shoulders. Figure 7 shows the patterns of neck location (x-axis) in pixels toward the camera for three groups of learners; locations for other body key points followed relatively similar patterns. The maximum neck distance from the camera that could be detected was 640 pixels (half of 1280 pixels) because the resolution of our cameras was 720p. Figure 7 shows that learners who received high satisfaction scores (green) sit closer toward the camera (they had closer distances to their partners) over the collaborative coding sessions, compared to learners who received average (red) and low (blue) satisfaction scores. Additionally, learners who received high satisfaction scores (green) had lower neck location variance over time, compared to learners who received average (red) and low (blue) satisfaction scores.

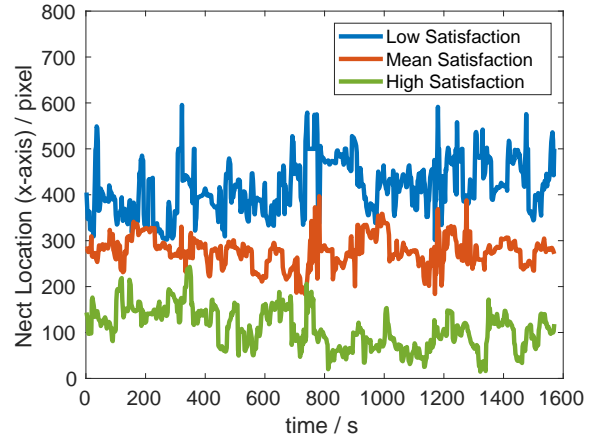


Figure 7: Neck Location (x-axis) in Pixels from OpenPose

The findings from Figure 6 and Figure 7 were aligned with previous literature that found learners’ perceived social presence and proximity significantly impacted their satisfaction, as well as group performance during collaborative learning [8, 36]. For example, a study conducted by So et al. [46] revealed that learners’ perceptions of physical proximity and psychological aspects of distance were both important factors in their reported satisfaction with their partner. In another similar study conducted by Spikol et al. [47], the

authors found that the distances between learners’ faces and between learners’ hands were two strong indicators of task performance when groups of college students were engaged in open-ended collaborative tasks.

7.2 RQ 2: Does multimodal feature fusion improve peer satisfaction prediction compared to the best-performing unimodal model?

In this study, experiment results in Table 3 from several multimodal models indicated that although using multimodal features (Head Position (x-axis) combined with Body Key Points) yielded lower *MAE* than the best-performing unimodal feature, there was no significant performance advantage of using multimodal over unimodal features. The potential reason may be that both head position and body key points represented learners’ spatial locations; therefore, combining these two unimodal features did not add extra useful information to predict peer satisfaction. Therefore, each of these two unimodal features alone could be used to predict peer satisfaction. However, the head pose feature extraction process with OpenFace was faster than OpenPose. OpenPose was computationally demanding, and required GPU acceleration to perform the body key points detection. Therefore, OpenFace may be a more practical feature extraction choice over OpenPose when deploying real-time learning support systems.

7.3 Implications from comparing different model architectures

The experimental results comparing performance of different model architectures showed that the three different sequential models (RNN, LSTM, and GRU) had similar peer satisfaction prediction accuracy; in addition, non-linear regression models yielded lower *MAE* than linear regression models. These results have a few practical implications for researchers in the educational data mining community seeking to conduct similar studies with the methodology presented in this study.

Although sequential models were able to represent the sequential nature of utterance-level features, the comparison between different sequential models (RNN, LSTM, and GRU) did not reflect significant performance differences. Given that GRU usually has a faster training speed than LSTM and RNN due to its simpler cell structure, GRU could be a better choice over RNN or LSTM for similar tasks. In addition, the comparison between different activation functions (*linear* and *sigmoid*) showed that the *sigmoid* regression model yielded lower *MAE* and provided more numerical stability during testing than the *linear* model. The reason may be that the satisfaction scores predicted in this study only ranged from 1 to 5, so the constrained output value range of the *sigmoid* function could better avoid large error values during training. On the contrary, there was no mechanism to prevent the *linear* activation function from predicting out-of-range satisfaction scores.

7.4 Limitations

The current work has several important limitations. First, we only studied peer satisfaction in the context of co-located pair programming, and analyzed recordings collected from

a relatively small corpus with 44 middle school learners; therefore, the predictive features found in this paper may not generalize well to group collaboration involving three or more team members, or to learners in other populations or learning environments, such as adults or online learning. Second, the LSTM-based feature learning process was black-box, which makes it relatively difficult to interpret what predictive information was learned from each unimodal feature. Finally, the effectiveness of video-derived features identified in this study relies heavily on the correct setup of the video recording process. Our dataset was collected from a natural and active classroom setting, and thus, OpenFace sometimes failed to detect both learners’ faces when they were not directly facing the camera, or in the case of occlusion. Even though we used wide-angle camera lenses for video recording student interactions, there were some cases in which some students were sometimes out of the recording range.

8. CONCLUSION AND FUTURE WORK

Learners’ satisfaction toward their partners plays a crucial role in group performance and learning outcomes. If intelligent systems could automatically predict peer satisfaction during collaboration, they could provide timely scaffolding for better learning experiences. In this paper, we investigated automatic prediction of peer satisfaction by analyzing 44 middle school learners’ collaborative dialogues. We compared a set of state-of-the-art multimodal learning analytics techniques with linguistic, acoustic-prosodic, and visual features extracted from students’ interactions. The experimental results revealed two significant predictors: head position and body location. Learners who had shorter head and body distances from their partners were more likely to receive higher peer satisfaction scores.

This study highlights several directions for future work. First, future work should examine the generalizability of the findings in this study using larger datasets, including data from online learning environments and multi-party interactions among groups of three or more learners. Second, although OpenFace and OpenPose support accurate detection of head pose and body pose, it remains challenging to integrate them into intelligent learning support systems for real-time analysis. Future work should investigate other methods and tools to detect learners’ pose features accurately and time-efficiently. Third, the satisfaction survey was administered post-hoc in this study. Future work could investigate potential variations in students’ attitudes that may occur during the ongoing collaborative process. Finally, it is important to investigate how intelligent systems can most effectively deliver feedback to learners during collaborative learning process. As we move toward predicting peer satisfaction in real time, we will be able to build and investigate systems that can significantly improve learners’ collaborative learning experience in classrooms.

9. ACKNOWLEDGMENTS

This research was supported by the National Science Foundation through grant DRL-1640141. Any opinions, findings, conclusions, or recommendations expressed in this report are those of the authors, and do not necessarily represent the official views, opinions, or policy of the National Science Foundation.

10. REFERENCES

- [1] T. Baltrusaitis, A. Zadeh, Y. C. Lim, and L.-P. Morency. Openface 2.0: Facial behavior analysis toolkit. In *Proceedings of the 13th IEEE International Conference on Automatic Face & Gesture Recognition*, pages 59–66. IEEE, 2018.
- [2] P. Blikstein. Multimodal learning analytics. In *Proceedings of the 3rd International Conference on Learning Analytics and Knowledge*, pages 102–106, 2013.
- [3] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7291–7299, 2017.
- [4] M. Celepkolu and K. E. Boyer. The importance of producing shared code through pair programming. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, pages 765–770, 2018.
- [5] M. Celepkolu and K. E. Boyer. Predicting student performance based on eye gaze during collaborative problem solving. In *Proceedings of the Group Interaction Frontiers in Technology*, pages 1–8. 2018.
- [6] M. Celepkolu and K. E. Boyer. Thematic Analysis of Students’ Reflections on Pair Programming in CS1. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education - SIGCSE ’18*, pages 771–776. ACM, 2018.
- [7] M. Celepkolu, D. A. Fussell, A. C. Galdo, K. E. Boyer, E. N. Wiebe, B. W. Mott, and J. C. Lester. Exploring middle school students’ reflections on the infusion of cs into science classrooms. In *Proceedings of the 51st ACM technical symposium on computer science education*, pages 671–677, 2020.
- [8] S. W. Chae. Perceived proximity and trust network on creative performance in virtual collaboration environment. *Procedia Computer Science*, 91:807–812, 2016.
- [9] C.-L. Chan, J. J. Jiang, and G. Klein. Team task skills as a facilitator for application and development skills. *IEEE Transactions on Engineering Management*, 55(3):434–441, 2008.
- [10] S. A. Chowdhury, E. A. Stepanov, G. Riccardi, et al. Predicting user satisfaction from turn-taking in spoken conversations. In *INTERSPEECH 2016*, pages 2910–2914, 2016.
- [11] B. Cimatti. Definition, development, assessment of soft skills and their role for the quality of organizations and enterprises. *International Journal for Quality Research*, 10(1):97–130, 2016.
- [12] A. D. Clarke and B. W. Tatler. Deriving an appropriate baseline for describing fixation behaviour. *Vision Research*, 102:41–51, 2014.
- [13] M. Cukurova, Q. Zhou, D. Spikol, and L. Landolfi. Modelling collaborative problem-solving competence with transparent learning analytics: is video data enough? In *Proceedings of the 10th International Conference on Learning Analytics and Knowledge*, pages 270–275, 2020.
- [14] I. Daoudi, E. Tranvouez, R. Chebil, B. Espinasse, and W. Chaari. An edm-based multimodal method for assessing learners’ affective states in collaborative crisis management serious games. In *Proceedings of the 13th International Conference on Educational Data Mining*, pages 596–600, 2020.
- [15] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [16] V. Echeverria, R. Martinez-Maldonado, and S. Buckingham Shum. Towards collaboration translucence: Giving meaning to multimodal group data. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pages 1–16, 2019.
- [17] F. Eyben, M. Wöllmer, and B. Schuller. Opensmile: the munich versatile and fast open-source audio feature extractor. In *Proceedings of the 2010 International Conference on Multimedia*, pages 1459–1462, 2010.
- [18] C. Forsyth, J. Andrews-Todd, and J. Steinberg. Are you really a team player? profiling of collaborative problem solvers in an online environment. In *Proceedings of the 13th International Conference on Educational Data Mining*, pages 403–408. ERIC, 2020.
- [19] B. Gambäck and U. K. Sikdar. Using convolutional neural networks to classify hate-speech. In *Proceedings of the 1st Workshop on Abusive Language Online*, pages 85–90, 2017.
- [20] T. T. Goud, V. Smrithirekha, and G. Sangeetha. Factors influencing group member satisfaction in the software industry. In *Proceedings of the International Conference on Data Engineering and Communication Technology*, pages 223–230. Springer, 2017.
- [21] J. F. Grafsgaard, J. B. Wiggins, K. E. Boyer, E. N. Wiebe, and J. C. Lester. Automatically recognizing facial indicators of frustration: a learning-centric analysis. In *2013 Humaine Association Conference on Affective Computing and Intelligent Interaction*, pages 159–165. IEEE, 2013.
- [22] L. Hasler-Waters and W. Napier. Building and supporting student team collaboration in the virtual classroom. *Quarterly Review of Distance Education*, 3(3):345–52, 2002.
- [23] N. Houssami, P. Macaskill, M. L. Marinovich, J. M. Dixon, L. Irwig, M. E. Brennan, and L. J. Solin. Meta-analysis of the impact of surgical margins on local recurrence in women with early-stage invasive breast cancer treated with breast-conserving therapy. *European Journal of Cancer*, 46(18):3219–3232, 2010.
- [24] K. Huang, T. Bryant, and B. Schneider. Identifying collaborative learning states using unsupervised machine learning on eye-tracking, physiological and motion sensor data. *Proceedings of The 12th International Conference on Educational Data Mining*, pages 318–323, 2019.
- [25] E. Kapp. Improving student teamwork in a collaborative project-based course. *College Teaching*, 57(3):139–143, 2009.
- [26] G. A. Katuka, R. T. Bex, M. Celepkolu, K. E. Boyer, E. Wiebe, B. Mott, and J. Lester. My partner was a good partner: Investigating the relationship between dialogue acts and satisfaction among middle school computer science learners. In *Proceedings of the 14th*

- International Conference on Computer-Supported Collaborative Learning*, pages 51–58. International Society of the Learning Sciences, 2021.
- [27] J. Kim, Y. Kwon, and D. Cho. Investigating factors that influence social presence and learning outcomes in distance higher education. *Computers & Education*, 57(2):1512–1520, 2011.
- [28] H.-Y. Ku, H. W. Tseng, and C. Akarasriworn. Collaboration factors, teamwork satisfaction, and student attitudes toward online collaborative learning. *Computers in Human Behavior*, 29(3):922–929, 2013.
- [29] Q. P. Law, H. C. So, and J. W. Chung. Effect of collaborative learning on enhancement of students’ self-efficacy, social skills and knowledge towards mobile apps development. *American Journal of Educational Research*, 5(1):25–29, 2017.
- [30] R. Liu, J. Davenport, and J. Stamper. Beyond log files: Using multi-modal data streams towards data-driven kc model improvement. *Proceedings of the 9th International Conference on Educational Data Mining*, pages 436–441, 2016.
- [31] C. N. Loes and E. T. Pascarella. Collaborative learning and critical thinking: Testing the link. *The Journal of Higher Education*, 88(5):726–753, 2017.
- [32] M. Madaio, R. Lasko, A. Ogan, and J. Cassell. Using temporal association rule mining to predict dyadic rapport in peer tutoring. *Proceedings of the 10th International Conference on Educational Data Mining*, pages 318–323, 2017.
- [33] J. Malmberg, S. Järvelä, J. Holappa, E. Haataja, X. Huang, and A. Siipo. Going beyond what is visible: What multichannel data can reveal about interaction in the context of collaborative learning? *Computers in Human Behavior*, 96:235–245, 2019.
- [34] R. W. Mee and T. C. Chua. Regression toward the mean and the paired sample t test. *The American Statistician*, 45(1):39–42, 1991.
- [35] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.
- [36] S. Molinillo, R. Aguilar-Illescas, R. Anaya-Sánchez, and M. Vallespín-Arán. Exploring the impacts of interactions, social presence and emotional engagement on active collaborative learning in a social web-based environment. *Computers & Education*, 123:41–52, 2018.
- [37] S. L. Pugh, S. K. Subburaj, A. R. Rao, A. E. Stewart, J. Andrews-Todd, and S. K. D’Mello. Say what? automatic modeling of collaborative problem solving skills from student speech in the wild. *Proceedings of The 14th International Conference on Educational Data Mining*, pages 55–67, 2021.
- [38] Pydub. <https://github.com/jiaaro/pydub>.
- [39] I. Radu, E. Tu, and B. Schneider. Relationships between body postures and collaborative learning states in an augmented reality study. In *Proceedings of the 21st International Conference on Artificial Intelligence in Education*, pages 257–262. Springer, 2020.
- [40] R. Rajendran, A. Kumar, K. E. Carter, D. T. Levin, and G. Biswas. Predicting learning by analyzing eye-gaze data of reading behavior. *Proceedings of the 11th International Conference on Educational Data Mining*, pages 455–461, 2018.
- [41] J. M. Reilly and B. Schneider. Predicting the quality of collaborative problem solving through linguistic analysis of discourse. *Proceedings of The 12th International Conference on Educational Data Mining*, pages 149–157, 2019.
- [42] Rev. <https://www.rev.com/>.
- [43] B. Schneider, K. Sharma, S. Cuendet, G. Zufferey, P. Dillenbourg, and R. Pea. Leveraging mobile eye-trackers to capture joint visual attention in co-located collaborative learning groups. *International Journal of Computer-Supported Collaborative Learning*, 13(3):241–261, 2018.
- [44] J. L. Schultz, J. R. Wilson, and K. C. Hess. Team-based classroom pedagogy reframed: The student perspective. *American Journal of Business Education*, 3(7):17–24, 2010.
- [45] A. J. Sinclair and B. Schneider. Linguistic and gestural coordination: Do learners converge in collaborative dialogue?. *Proceedings of The 14th International Conference on Educational Data Mining*, pages 431–438, 2021.
- [46] H.-J. So and T. A. Brush. Student perceptions of collaborative learning, social presence and satisfaction in a blended learning environment: Relationships and critical factors. *Computers & Education*, 51(1):318–336, 2008.
- [47] D. Spikol, E. Ruffaldi, L. Landolfi, and M. Cukurova. Estimation of success in collaborative learning based on multimodal learning analytics features. In *2017 IEEE 17th International Conference on Advanced Learning Technologies (ICALT)*, pages 269–273, 2017.
- [48] A. E. Stewart, Z. Keirn, and S. K. D’Mello. Multimodal modeling of collaborative problem-solving facets in triads. *User Modeling and User-Adapted Interaction*, 31(4):713–751, 2021.
- [49] A. E. Stewart, Z. A. Keirn, and S. K. D’Mello. Multimodal modeling of coordination and coregulation patterns in speech rate during triadic collaborative problem solving. In *Proceedings of the 20th ACM International Conference on Multimodal Interaction*, pages 21–30, 2018.
- [50] W. Wei, S. Li, S. Okada, and K. Komatani. Multimodal user satisfaction recognition for non-task oriented dialogue systems. In *Proceedings of the 2021 International Conference on Multimodal Interaction*, pages 586–594, 2021.
- [51] R. M. Zeitun, K. S. Abdulqader, and K. A. Alshare. Team satisfaction and student group performance: A cross-cultural study. *Journal of Education for Business*, 88(5):286–293, 2013.
- [52] X. Zhang, Y. Meng, P. O. de Pablos, and Y. Sun. Learning analytics in collaborative learning supported by slack: From the perspective of engagement. *Computers in Human Behavior*, 92:625–633, 2019.

Going Deep and Far: Gaze-Based Models Predict Multiple Depths of Comprehension During and One Week Following Reading

Megan Caruso, Candace E. Peacock, Rosy Southwell, Guojing Zhou, Sidney K. D'Mello
University of Colorado Boulder

megan.caruso@colorado.edu; peacock.candace@gmail.com; roso8920@colorado.edu;
guojing.zhou@colorado.edu; sidney.dmello@colorado.edu

ABSTRACT

What can eye movements reveal about reading, a complex skill ubiquitous in everyday life? Research suggests that gaze can reflect short-term comprehension for facts, but it is unknown whether it can measure long-term, deep comprehension. We tracked gaze while 147 participants read long, connected, informative texts and completed assessments of rote (factual) and inference comprehension (connecting ideas) while reading a text, after reading a text, after reading five texts, and after a seven-day delay. Gaze-based student-independent computational models predicted both immediate and long-term rote and inference comprehension with moderate accuracies. Surprisingly, the models were most accurate for comprehension assessed after reading all texts and predicted comprehension even after a week-long delay. This shows that eye movements can provide a lens into the cognitive processes underlying reading comprehension, including inference formation, and the consolidation of information into long-term memory, which has implications for intelligent student interfaces that can automatically detect and repair comprehension in real-time.

Keywords

Reading comprehension, Eye movements, Machine Learning, Long-term comprehension, Comprehension depth.

1. INTRODUCTION

Reading comprehension is the extraction of meaning from text. This activity takes place many times a day, whether reading the news, absorbing technical information at school or work, or reading a novel for pleasure. Difficulty in reading comprehension can slow the progression of such activities, and comprehension failures can lead to misunderstandings and inaccuracies. The rise of computerized reading via e-books, the Internet, and other media opens up the exciting possibility of intelligent interfaces that can track reading comprehension as it unfolds based on measurable signals (behaviors) from the reader [13, 51].

Eye-gaze is perhaps one attractive signal to explore because it provides a lens into cognitive processes [30, 48] and it can be passively and noninvasively recorded. In particular, there is a long history of using eye gaze in student-models of cognitive, affective, and social

M. Caruso, C. Peacock, R. Southwell, G. Zhou, and S. D'Mello. Going deep and far: Gaze-based models predict multiple depths of comprehension during and one week following reading. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 145–157, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852998>

processes during learning [8, 15, 61]. In the context of reading comprehension researchers have developed automatic models for skimming [3] and mind wandering (zone outs) [19] detection. These models have also been used for real-time intervention. For example, Mills et al. [41] designed an attention-aware reading intervention that prompted participants to re-read sections of text based on a real-time gaze-based model of mind wandering [19] and found this to improve reading comprehension. In addition, real-time modifications can be made to text content, such as adapting the text to be easier when comprehension difficulty is detected [53], or enabling gaze-contingent actions such as presenting a glossary for technical terms [3].

Whereas these examples focus on adapting the reading interface based on ongoing comprehension processes, such as mind wandering or comprehension difficulty, another possibility is to base adaptations on comprehension outcomes. For example, if gaze can be used to prospectively predict whether a student will comprehend a page or an entire text, adaptive interventions can be designed to address such deficits at their onset. Such a system would entail developing a model to monitor comprehension outcomes from gaze as a first step, a possibility we explore here. Specifically, we examine whether machine-learned models of gaze can be used to predict different types of comprehension outcomes (factual vs. those requiring inferencing) assessed at different time intervals (during a text, after a text, after multiple texts, and greater than a week). In addition to potential applications, the present research advances the empirical knowledge base of eye movements in reading comprehension, and to our best knowledge, is the first such study.

2. BACKGROUND AND RELATED WORK

2.1 Reading Comprehension

Most theories of reading posit that it involves hierarchically interacting levels of processing - from the sub-lexical and lexical levels [25] and the access of word-level meaning [45] to the formation of a literal then a more abstract meaning-based encoding of the text at the sentence-level (Figure 1 links a-c; [59]). Higher-order (or deeper) processing incorporates elaborative inferences from prior knowledge (Figure 1 n) [32] and integration across multiple sections within the text and even between texts (bridging inferences), forming a situation (or mental) model (Figure 1 d-g) [24, 39]. These above shallow and deep comprehension processes unfold in parallel [33] and interact with one another to provide a cohesive narrative of text. Both are critical in that shallow, perceptual encoding of information is important to construct the mental representations to support inferences from the text, and inferences are important to bridge ideas in the text into a cohesive narrative (Figure 1d; [34]).

It is therefore critical to develop models of comprehension that account for multiple levels of comprehension, which we address by focusing on varying depths of comprehension, such as rote (knowledge of factual content from text) and inference (deep) comprehension (for examples, see Table 2).

Comprehension also unfolds over timescales of milliseconds, seconds, minutes, hours, and beyond. For instance, reprocessing of remembered information can occur in milliseconds without re-fixating the part of the text that was not correctly encoded initially [6, 40], whereas comprehension of earlier sentences affects encoding of later content and vice versa (e.g., via bridging inferences [34]), a process that may unfold over seconds or even minutes. Further, memory traces acquired during reading one text may interfere with another [63] prior to being consolidated into long-term memory during sleep [43]. Thus, it is unclear if eye movements captured during the initial encoding of text (Figure 1 b,h) will be useful to predict comprehension at later stages after these intervening processes have unfolded (Figure 1 i-m). We address this by investigating the link between gaze and comprehension assessed at multiple time points.

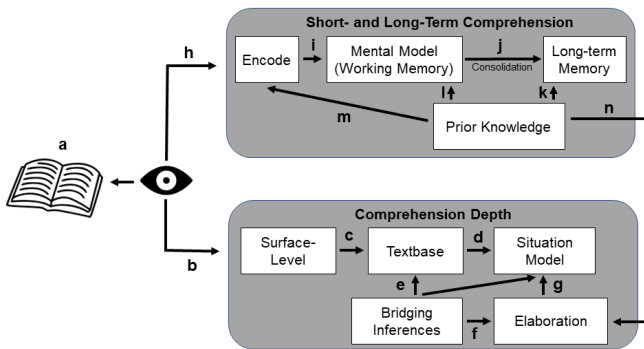


Figure 1. An overview of the process of reading comprehension.

2.2 Eye Movements During Reading

Theories of gaze control during reading shed light on the mechanisms linking cognitive processes and measurable gaze features [18, 50], particularly for lower-level lexical processes. For example, eye movements are determined to some extent by text properties, where fixations are shorter on more frequent [47] and shorter [49] words. But eye movements are also influenced by higher-level comprehension [5], for example the same word may be fixated for longer if it is contextually surprising versus expected

[38] or regressive eye movements can reconcile disparities in comprehension [9], such as in the case of ambiguous sentences [23] (but this can also occur covertly without a regression [7]).

Gaze has also been linked to several processes that support comprehension. In particular, attentional lapses (mind wandering) which are negatively associated with comprehension [14, 54] have been linked with fewer and longer fixations (see brief review by Faber et al. [20]). Similarly, skim reading manifests as fewer fixations [37] and fewer regressive saccades [42], but in contrast to mind wandering, these fixations are shorter than for normal reading [56].

Researchers have leveraged these findings to develop gaze-based models of reading comprehension as noted in Table 1. Whereas there have been attempts to model comprehension from other signals, such as facial expressions [58], we focus on gaze here. In general, most studies compute gaze statistics aggregated over time to give “global” measures at the level of an entire page, passage, or reader for use as features in shallow machine learning classifiers (e.g., Random Forest). For example, D’Mello et al. [16] used linear regression models to predict responses to multiple-choice questions targeting rote (factual) information of the text interspersed during reading trained on a small number of global gaze features grounded in the experimental literature. Similarly, Copeland and colleagues trained shallow neural networks to predict comprehension scores from gaze [10–12].

Deep neural networks are capable of modeling gaze behavior [35, 62], so it is plausible that an end-to-end system could be designed to predict comprehension. To this point, Ahn et al. [1] used convolutional neural networks (CNN) and long short-term memory (LSTM) models to model several reading comprehension metrics (passage- and participant-level comprehension, perceived difficulty, English language skill) from raw fixation-level data (location, duration, and pupil diameter) but their model performance was scarcely above chance and the results did not generalize to new readers.

To this point, a vast majority of studies do not provide evidence of generalizability to new people where data from the same participants are in either the training or test set (but not both). Further, as evident in Table 1, almost all studies focus on rote or inference comprehension or a combination of the two, which makes it difficult to compare the performance of gaze-based models for either type; comprehension is almost always assessed during reading or immediately after, but never after a longer delay when interference and memory consolidation processes unfold.

Table 1: Review of gaze-based computational models of reading comprehension

Study	Comprehension Type	Assessment Time	Participant-level Generalization	Small, predetermined feature set
Copeland & Gedeon, 2013	Rote	During reading	No	Yes
Copeland et al., 2014	Rote	Immediately after reading	No	Yes
Martínez-Gómez & Aizawa, 2014	Rote	Immediately after each text/section	Yes	Yes
Wallot et al., 2015	Rote	Immediately after reading	No	Yes
Copeland et al., 2016	Rote	Immediately after each text/section	No	Yes
Ahn et al., 2020	Rote and inference	Immediately after each passage	No	NA
D’Mello et al., 2020	Rote	During reading	Yes	Yes
Southwell et al., 2020	Rote (2 studies) and inference (1 study)	Roughly 30 minutes after reading	Yes	Yes

2.3 Current Study: Contribution & Novelty

As reviewed above, there is reason to suggest that gaze can provide an important signal to automatically measure comprehension during reading. However, despite some initial attempts towards this goal (Table 1), substantial items remain including: (1) differentiating the predictive value of gaze on different depths of comprehension; and (2) different time onsets from the initial reading of the text (where gaze data is acquired) and when comprehension is assessed; (3) developing models that generalize to new students, and (4) testing whether deep sequence learning models can improve comprehension prediction above standard classifiers.

We addressed these issues by testing whether gaze could be used to predict rote and inference comprehension (#1 above) assessed at four time points (during reading, after each text, after all texts, and at least seven days following reading; #2), using a large dataset of eye movements recorded as 147 participants read five long expository texts. Random forest models were used to evaluate whether a broad set of page-level gaze features (109 features total) could be used to detect reading comprehension on each page across depth and time by comparing them to two baseline models. We also tested a long-short-term memory (LSTM) deep neural network to examine whether temporal sequences of fixations could improve predictive accuracy (#4). Critically, participant-level cross-validation was used to increase the validity of the model on new participants (#3).

It should be noted that the present goal is more scientific in nature – to investigate the relationship between eye tracking and reading comprehension outcomes – rather than application oriented. As such, although we used a machine-learning predictive modeling approach [17], the goal was to use the models to investigate the question of the link between gaze and comprehension depth and durability (persistence across time) rather than to engineer the most predictive model. For this reason, we largely restricted the feature space to high-level eye gaze features and some contextual variables but did not include information on textual content and difficulty of the assessment items.

3. METHOD

3.1 Data Collection

Data was collected as part of a larger study investigating neurophysiology during reading comprehension. Only aspects germane to the present study are presented here. The data analyzed here have not been previously published.

Participants (N=147, age 23±6 years, 67% female, 1% other) were students from a large public University in the Western US. Participants were paid \$20 per hour plus \$10 for a follow-up survey via Amazon gift cards. All procedures were approved by the institution’s internal review board and all participants provided informed consent.

Binocular gaze was tracked using a high-resolution desktop-mounted eye tracker (SR Research EyeLink 1000+) with a sampling rate of 1000Hz. Stimuli were displayed on a 23.8”, 1920x1080 pixel display, and participants viewed the screen at a distance of ~90cm. A chin rest was used to minimize movement during the study.

Participants read five expository texts of around 1000 words each, where a single text was split into 10 pages. Each text was on the topic of behavioral research methods: Bias, Hypothesis, Casual

Claims, Validity, and Variables. The texts had a mean Flesch-Kincaid grade level (a measure of textual difficulty) of 13.2 indicating an advanced reading level [22] suitable for college students. Reading was self-paced in that participants pressed a key to advance to the next page but could not advance back to a previous page. On average, participants spent 5.5 minutes (1.8 SD) reading each text, for a total average reading time of 27.6 minutes (9.2 SD).

Reading comprehension was assessed via: 1) Rote items - four-alternative forced-choice item targeting factual knowledge explicitly presented in the text); 2) Inference items - a statement which is either a valid or false inference from the text, which the participant identified as ‘true’ or ‘false’ (see Table 2 for examples). Rote and inference questions were developed by researchers led by an instructor of a behavioral research methods undergraduate course. They were then piloted and refined using Mechanical Turk in order to calibrate the difficulty to the appropriate level. Items with less than 20% accuracy or greater than 80% accuracy were reexamined and either discarded or adapted based on patterns of responses. The final assessments and texts included in the study were tested on 10 unique participants per text.

Table 2: Examples of a rote question and inference items

Text	Question
“Occam’s razor, also called the principle of parsimony, teaches that hypotheses introduced to explain relationships should be as parsimonious as possible. We ‘cut away’ what is superfluous.”	(Rote Question) What is Occam’s Razor? A) The process by which we search for the simplest explanation for an observation (correct); B) The process by which we search for connections between facts; C) A tool used by those who search for truth; D) A book by the philosopher William of Occam
Sentence S3: Internal validity refers to whether the relationship between the variables is free of confounds Sentence S5: Content validity refers to the extent to which a measure represents all facets of a given construct.	(Inference Question) True or False: Internal validity is not a prerequisite for content validity. False (correct response because for a test to separately measure all facets of a construct (S5) it must be able to identify relationships free of confounds (S3)).

Both assessment items occurred at four time points: (A1) “during text” occurred immediately after reading the corresponding page, (A2) “after each” occurred after each individual text was completed; (A3) “after all” occurred once all five texts were read; and (A4) “delay” occurred a minimum of seven days after the reading session (Figure 2; median completion time 8.0 days, mean completion time 11.3 days after reading). The assessment items were linked to content covered on a particular page such that gaze on that page could be associated with a corresponding assessment item (Figure 2). At each time point, each participant received assessments corresponding to a randomized subset of two pages for each text from a pool of questions common to the A1, A2 and A3 assessments (Figure 2) but without overlap (e.g., if a page was selected for A1, then it could not be used for A2 and A3). A4 assessments were selected from a different pool of questions than A1-A3.

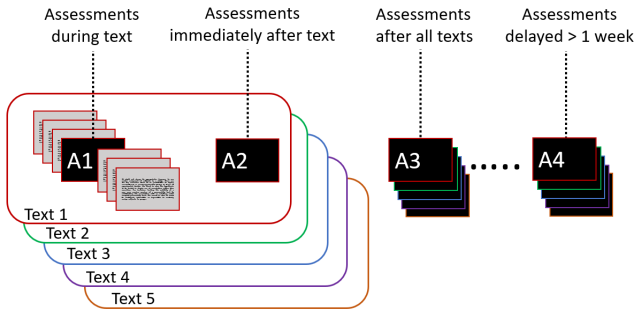


Figure 2. Four different timings of assessment questions. Black boxes indicate assessments, grey boxes indicate example pages within text, and colors indicate the five individual texts (and subsequently the corresponding text of the assessment questions).

3.2 Data Processing and Feature Extraction

Gaze data was processed with EyeLink’s event detection algorithm, using a velocity threshold of 30°/s and an acceleration threshold of 9500°/s². The right eye was used in the analyses if available, otherwise the left eye was used. No manual alignment of eye movements was done to address eye tracking errors as this would not be possible in a real-time application. Further, a pilot study comparing features (see Table 3) extracted from aligned vs. unaligned tested on the A1 (‘during text’) assessments yielded highly similar results.

Fixations and saccades greater than the 99th percentile across participants were removed to account for mis-parsed fixations and saccades (i.e., saccade amplitudes greater than 20°, durations above 600 ms, peak velocity below 5°/s or above 800°/s, distances over 1000 pixels, and fixation durations below 40 ms and above 3000 ms). The first and last fixations on a page were removed as these likely corresponded to orienting rather than reading behaviors.

Table 3. Gaze Feature Definitions

Feature	Description
Fixation Duration	Duration of a fixation in milliseconds
Fixation Count	Number of fixations on a page
Saccade Amplitude	Degrees of visual angle the eye travels during a saccade
Saccade Velocity	Saccade amplitude divided by saccade duration
Saccade Distance	Euclidian distance between saccade start and end points
Saccade Duration	Duration in milliseconds of a saccade
Saccade Relative Angle	Acute angle between the line segments of two saccades
Horizontal Saccade Angle	Angle between a saccade and the horizontal axis
Pupil Diameter (Z)	Diameter of the pupil, z-scored within-participant
Fixation Dispersion	Root mean square of distance from each fixation to the mean fixation position on a page
Horizontal Saccade Prop.	Proportion of saccades no greater than 30 degrees above or below the horizontal axis to the right or left
Forward Horizontal Saccade Prop.	Proportion of saccades no greater than 30 degrees above or below the horizontal axis only to the right
Blink Count	Number of blinks on a given page
Blink Duration	Duration in milliseconds of a blink
Dwell Time	Sum of the durations across all fixations that fell in a given interest area (i.e., the box bounding each word). Reflects the amount of time spent fixating on the words
IA Fixation Prop.	Total proportion of fixations that landed in a given interest area (e.g., 2 for word #12 “following” in Figure 3, 2/13 for the proportion)
Regression-In Prop.	Proportion of times an interest area was entered at the beginning of a regression. (e.g., 1 for #3 “start” in Figure 3, and 1/13 for the proportion)
Regression-Out Full Prop.	Proportion of times an interest area was regressed from (e.g., 2 for “experiment” #13 in Figure 3, and 2/3 for the proportion)
First Pass Regression-Out Prop.	Proportion of times an interest area was regressed from on the first pass (before having read any text past that interest area). (e.g., 1 for “experiment” #13 in Figure 3, and 1/13 for the proportion)
Regression Path Duration	Total time from when an interest area is fixated until it is exited to the right (also called go-past duration). This includes all the time spent regressing until that interest area is passed. (e.g., the sum of the first fixation on “experiment” #13 in addition to the sum of the 3 subsequent regressive fixations, and the last fixation on that same word in Figure 3).
Selective Regression Path Duration	Total gaze duration (duration of fixations and refixations) on an interest area before leaving the interest area to the right (e.g., 2 for the word “following” and the word “experiment” #13 in Figure 3).

The word “proportion” indicates these features were normalized by the total number of fixations on the given page. Prop. = Proportion

3.2.1 Gaze Features

Global (content-independent) gaze features (Table 3) were calculated as statistical functions over low-level features at the page-level (including min, max, mean, median, sum, skew, kurtosis, and standard deviation) and have been previously used to predict comprehension and mind wandering [4, 19, 55]. A second set of features captured the fixations corresponding to interest areas, which were rectangular boxes around individual words and punctuation computed with EyeLink Dataviewer (see example in Figure 3). The gaze models consisted of these 109 features plus three context features (see below). One goal of these features is to find participant-general patterns in gaze and comprehension, so to account for the variation in individual differences in fixation rates [26, 60], we normalized interest area features by the total number of fixations on a given page.

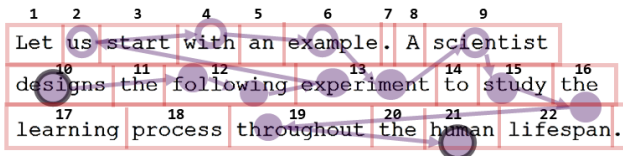


Figure 3. Hypothetical fixations ($n = 13$) and saccades on the text. The numbers indicate the interest area index for the word/punctuation they are above, the circles indicate fixations and the lines indicate saccades. The circles with darker outlines are the first and last fixation, for reference. The unfilled circles denote regressive fixations.

Table 4 includes summary statistics for a few key gaze features used in the present study, which are consistent with typical values observed during reading [46, 48]. The mean fixation duration was 210.30 ms, well in range of the 200-250 ms average during reading, and the mean saccade duration of 42.04 ms is close to the average saccade duration of 50 ms during reading [46]. Further, the average saccade amplitude in the study was 3.46 degrees, whereas the average during reading is reported to be 2 degrees [48]. This study does differ slightly in regressive fixations, as it is estimated that 10-15% of saccades are regressive during reading [46], where this study had 34%. There was also a slightly higher horizontal saccade proportional (95%) than previous studies (e.g., 85% in D’Mello et al. [16]).

Table 4. Gaze summary statistics computed over participants. IA = interest area.

Feature	M (SD)
Mean Saccade Amplitude	3.46 (0.45)
Total Scan Path Length	422.68 (105.61)
Fixation Dispersion	0.41 (0.03)
Mean Fixation Duration	210.84 (25.20)
Mean Saccade Duration	42.04 (8.94)
Horizontal Saccade Proportion	0.95 (0.03)
Mean Pupil Size (z)	0.11 (0.30)
Regression Fixation Proportion	0.34 (0.09)
IA Percent Visited	0.56 (0.08)
Proportion of Fixations in IAs	0.91 (0.06)
Mean IA Regression Path Duration	984.57 (450.88)
Mean Blink Duration	189.91 (229.87)

3.2.2 Baseline Models: Context Features, Shuffled Labels and Shuffled Fixation Events

Context features capture situational factors independent of gaze and were used as a baseline measure to gauge the added value of gaze features. They included reading rate (reading time divided by the number of characters on a page), text order, and the eye tracker calibration error. For a second baseline, we fit gaze models where comprehension scores were shuffled within each participant, preserving the distribution of the features, but breaking the temporal dependency between gaze and the comprehension. Additionally, the LSTM baseline models shuffled the units in the sequence of fixation events (see below), preserving the distribution of the features, but breaking the temporal dependency between fixation events.

3.3 Machine Learning Models

We chose Random Forest classifiers since they incorporate nonlinearity and interactivity among features and have good generalization properties. The random forest classifier was implemented in sklearn, with 100 estimators, minimum of 15 samples per leaf, and the maximum number of features set to the square root of the total number of features. The class weights of the models were balanced by setting the weights to be inversely proportional to the number of samples in each class. Note that no resampling was done on either the training or testing sets: setting the class weights to ‘balanced’ simply penalizes wrong predictions made on the minority class.

We also trained LSTM models (implemented in Keras) to examine to what extent the sequence of local fixation events can be used to predict reading comprehension. Each unit in the sequences represents a fixation event, which was described by five features: 1) fixation duration, 2) average horizontal (x) position, 3) average vertical (y) position, 4) average pupil size, and 5) the elapsed time since the end of last fixation. We also explored using other features, such as the position of the previous and next fixation, and the distance between the current and previous fixation, but this did not improve model performance. The maximum sequence length was set to 160 units, which is longer than 85% of the sequences. For those shorter than 160 units, 0s were filled at the beginning and for the longer ones, the last 160 units were kept. The LSTM network included a LSTM layer followed by two fully connected layers and used the binary cross entropy loss function. We tuned the following hyper-parameters: the number of hidden nodes in the LSTM layer (e.g., 8, 16, 32), the number of nodes in the fully connected layers (e.g., 8, 16, 32), batch size (e.g., 16, 32, 64), and dropout rate (e.g., 0, 0.2, 0.4). The hyper-parameters for each model were selected through a random search in 4-fold cross validation with 50% of data for training, 25% for validation, and 25% for testing.

3.4 Validation, Metrics, and Statistical Comparisons

We used four-fold cross-validation at the participant level to ensure generalizability to new participants [16]. Here, the dataset was randomly split into four folds, with the data from a given participant only being in a single fold. The process was repeated 10 times with a different random partitioning of the folds for each run. The same fold assignments were used to train the Random Forest gaze models and baseline models per run, but fold assignments were not preserved for the LSTM models as they were run in a different pipeline. For the random forest models, we only used participants who completed all assessments for a fair comparison across time ($N=122$). All participants (irrespective of whether they completed

all assessments) were used for the LSTMs to maximize the data needed for these data-intensive models (N=147). Because there was very little variability across runs, predictions were pooled for each participant from all runs prior to computing accuracy measures.

Performance was evaluated using the area under the precision-recall curve (AUPRC), which ranges from 0 to 1 with the ratio of true classes to total data (i.e., base rates) indicating baseline classification by guessing. The AUPRC was used because it is well-suited for class imbalance unlike the receiver operator curve (ROC) which may provide an overly optimistic view of model performance when classes are imbalanced [29]. AUPRCs were separately computed for each assessment type and time on a per-participant basis, resulting in eight values per participant per classification model.

We used linear mixed models [21] via the lmer package in R [2] to compare the percent improvement of AUPRCs over baseline ($100 * ((AUPRC - \text{base rate}) / \text{base rate})$). Mixed models are the recommended approach due to the repeated nature of the data (i.e.,

eight values per participant per model). Here, participant was included as an intercept-only random effect. We probed significant effects with the emmeans (estimated marginal means) package using a false discovery rate (FDR) adjustment for multiple comparisons and a two-tailed $p < .05$ significant criterion.

4. RESULTS

4.1 Observed comprehension differences across depth and time

We first considered how comprehension changed as a function of depth and time (Figure 4). To examine the extent to which each assessment measured the same construct, we first computed the proportion correct for each assessment for each participant. From here, we computed the Pearson correlation between the comprehension scores for each pairwise assessment (Table 5 upper diagonal). Overall, the average correlation was 0.23 and ranged from -0.13 to 0.54, suggesting that there was some overlap but also unique information in what each assessment measured.

Table 5. Pairwise Pearson correlations of comprehension scores and gaze model probabilities averaged over participants. The upper diagonal (white) contains the correlation between comprehension accuracies across depth and time. The lower diagonal (grey) contains the correlations between random forest gaze model probabilities across depth and time.

Assessment	Rote: During	Rote: After Each	Rote: After All	Rote: Delay	Inference: During	Inference: After Each	Inference: After All	Inference: Delay
Rote: During	-	0.33	0.42	0.35	0.38	0.13	0.08	0.11
Rote: After Each	0.19	-	0.54	0.37	0.26	0.23	0.17	0.11
Rote: After All	0.51	0.04	-	0.47	0.44	0.22	0.18	0.18
Rote: Delay	-0.01	0.18	-0.02	-	0.35	0.26	0.25	0.18
Inference: During	0.62	0.05	0.51	-0.03	-	0.17	0.16	0.14
Inference: After Each	0.02	0.21	-0.10	0.10	-0.02	-	0.20	0.10
Inference: After All	0.02	-0.01	0.16	0.02	0.00	0.00	-	-0.13
Inference: Delay	0.16	0.07	0.30	-0.10	0.18	-0.28	-0.07	-

Next, we examined how averaged participant-level comprehension measures varied as a function of depth and time using the following linear mixed-effects model: $\text{proportion correct} \sim \text{depth} * \text{time} + (1 | \text{participant})$. There were significant main effects ($ps < 0.01$) and interactions ($ps < 0.07$). For the main effect of comprehension depth, the rote assessment scores were significantly higher than the inference assessments ($B = 0.05, p = 0.01$). We then probed the significant interactions using emmeans. For rote comprehension, the mean score during reading was statistically equivalent to the score after reading each text ($p = 0.10$) which were both statistically greater ($p < 0.05$) than assessments after reading all the texts ($p > 0.05$) and at delay ($p > 0.05$), suggesting the following pattern: [During = After Each] > [After All = Delay]. This suggested that as people read, rote comprehension was stable but dropped upon completion of reading. Inference comprehension was stable across the reading session but dropped at delay with the pattern of significance: [During = After Each = After All] > Delay.

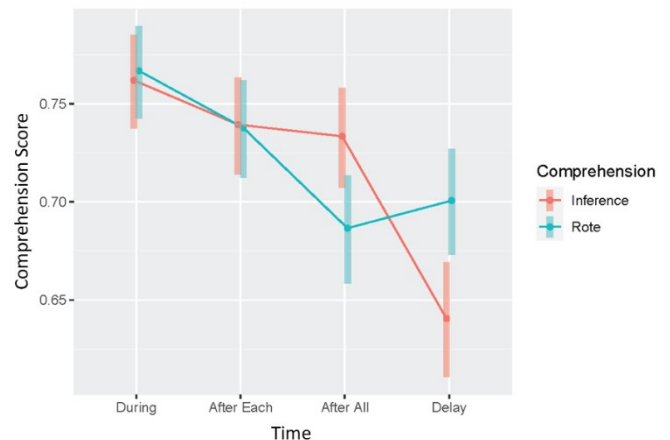


Figure 4. Predicted comprehension score from the mixed model as a function of depth and time. Error bands represent 95% confidence intervals.

4.2 Gaze Models can predict short and long-term reading comprehension

Gaze vs. shuffled (Random Forest and LSTM). Our first comparison examined whether the random forest gaze models performed significantly better than the shuffled models across depth and time (Figure 5): percent improvement \sim model type (shuffled vs. gaze)*depth*time + (1|participant). Overall, there was a significant interaction between model type and time ($p < 0.001$). All other interactions and main effects were non-significant (all $ps > 0.05$). When probing the model type x time interaction, we found that the gaze models outperformed the shuffled models for all cases (Figure 5; $ps < 0.001$) except for the “after each” model ($p = 0.37$), though the trend was in the expected direction. When we repeated the analysis for the LSTM model, the interactions and main effects of interest were non-significant (all $ps > 0.28$), indicating that they performed at chance in all cases and was indistinguishable from the shuffled model. Given the chance performance of the LSTMs, we focus on the random forest model results.

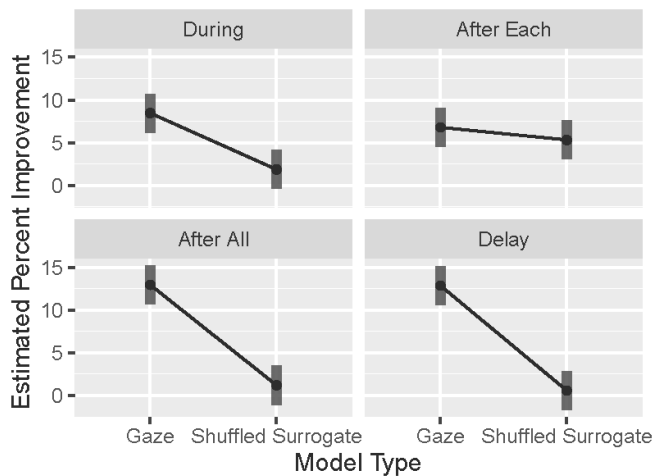


Figure 5. Gaze versus shuffled models. Comparison of the percent improvement for the gaze and shuffled models across time. Error bands represent 95% confidence intervals.

Gaze vs. context (Random Forest only). Our second analysis evaluated whether the random forest gaze models performed significantly better than the context models across depth and time (Figure 6): percent improvement \sim model type (gaze vs. context)*depth*time + (1|participant). Overall, there was a model by time interaction ($p = 0.002$), indicating that the gaze model outperformed the context model for all cases ($ps < 0.05$) except for the “after each” model ($p = 0.14$), though the trend was in the expected direction. Furthermore, there were no significant differences across time for the context model, as expected. However, for gaze, there were differences across time (Figure 6b). Specifically, the model performance for comprehension assessed after reading all texts was statistically equivalent to the delay model ($p = 0.90$) which was statistically higher than for assessments during reading ($p = 0.02$) which was in turn equivalent to assessments after each text ($p = 0.34$), suggesting the following pattern: [After All = Delay] $>$ [During = After Each]. There were no significant main effects nor interactions for comprehension depth, indicating similar patterns for rote and inference comprehension items.

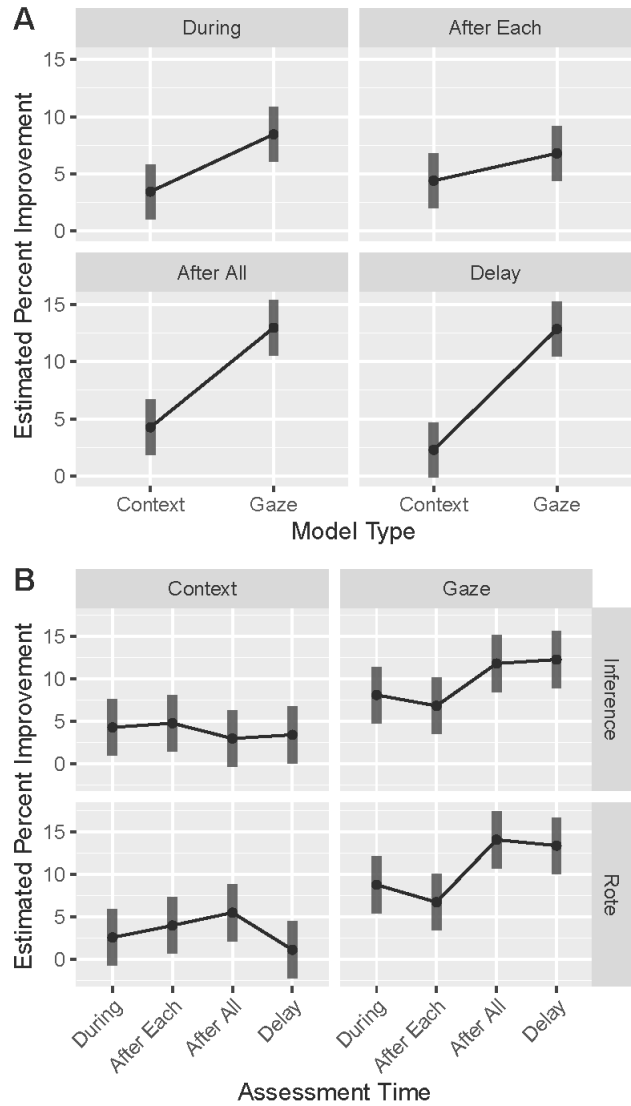


Figure 6. Comparison for percent improvement of gaze vs. context (A) across time and (B) across depth and time. Error bands represent 95% confidence intervals.

Table 6. Mean AUPRCs with 95% CIs computed over participants for the Random Forest models

Assessment	Gaze	Context	Base Rate
Rote: During	0.81 [0.80, 0.81]	0.77 [0.77, 0.78]	0.76
Rote: After Each	0.76 [0.76, 0.76]	0.74 [0.74, 0.75]	0.72
Rote: After All	0.75 [0.75, 0.76]	0.71 [0.71, 0.72]	0.68
Rote: Delay	0.75 [0.75, 0.75]	0.68 [0.68, 0.69]	0.68
Inference: During	0.80 [0.80, 0.81]	0.78 [0.78, 0.78]	0.75
Inference: After Each	0.78 [0.78, 0.78]	0.76 [0.76, 0.76]	0.73
Inference: After All	0.80 [0.80, 0.80]	0.74 [0.74, 0.75]	0.73
Inference: Delay	0.71 [0.70, 0.71]	0.65 [0.65, 0.65]	0.63

Model discrimination. To understand to what extent the random forest gaze model predictions were generally picking up on the same comprehension constructs, we computed the average model probabilities for each participant and assessment and correlated them (Table 5, grey diagonal). Overall, the average correlation was 0.10 and ranged from -0.28 to 0.62, suggesting that the models were

discriminating among the different comprehension assessments, with the highest correlation being between Rote During and Inference During, suggesting the model might be picking up on similar gaze patterns for both ‘During’ assessments.

4.3 Feature Analyses (Random Forest only)

We examined impurity-based feature importances from the Random Forest model to determine which aspects of gaze were most predictive of comprehension and whether the predicted features differed across comprehension types and delays. Here, we computed the correlation of the feature importances between each pairwise assessment for each run and then averaged across runs (Table 7). The grand average correlation excluding the diagonals was 0.15, indicating some overlap of feature importances across depth and time despite the broad range of features. Overall, the rote measures showed a mean $r_s = 0.23$, with a range from -0.01 to 0.35 (light grey in Table 7), with rote comprehension during reading showing the lowest correlations with the other rote assessments ($r_s = -0.01, 0.06$, and 0.16 compared to $r_s = 0.27, 0.30$, and 0.35 for the other rote assessments). Conversely, the inference comprehension measures showed a wide range of correlations with each other (from $r_s = -0.14$ to 0.57). This is in part due to the delayed inference assessments showing negative correlations with all other inference assessments ($r_s = -0.14, -0.05, -0.10$), while all other correlations

were positive, with inferencing assessed during reading and after reading all texts showing the highest correlation ($r = 0.57$). Correlations among rote and inference assessments showed a mean $r_s = 0.15$, with a range from -0.10 to 0.57 and the strongest correlations among the delayed rote assessments and inferencing assessed during the session (r_s from 0.26 to 0.57). Thus, the pattern of inter-associations among feature importances is mixed.

We then assessed which features were most important to comprehension across depth and time. To this end, we first computed the overall mean feature importances for each run across all assessments and averaged across runs. We then grouped and averaged each set of statistical features together to assess relative importance of each feature type. For example, for the “fixation duration” feature set, the feature importances for the eight statistical features of fixation duration were averaged together. We found that the top four feature categories were: the selective regression path duration (average importance = 0.0103), regression fixation proportion (average importance = 0.0097), calibration error (average importance = 0.0094), and interest area dwell time (average importance = 0.0094). Although these features were the most important, it appeared that all features contributed to model performance as they were all greater than zero (Figure 7).

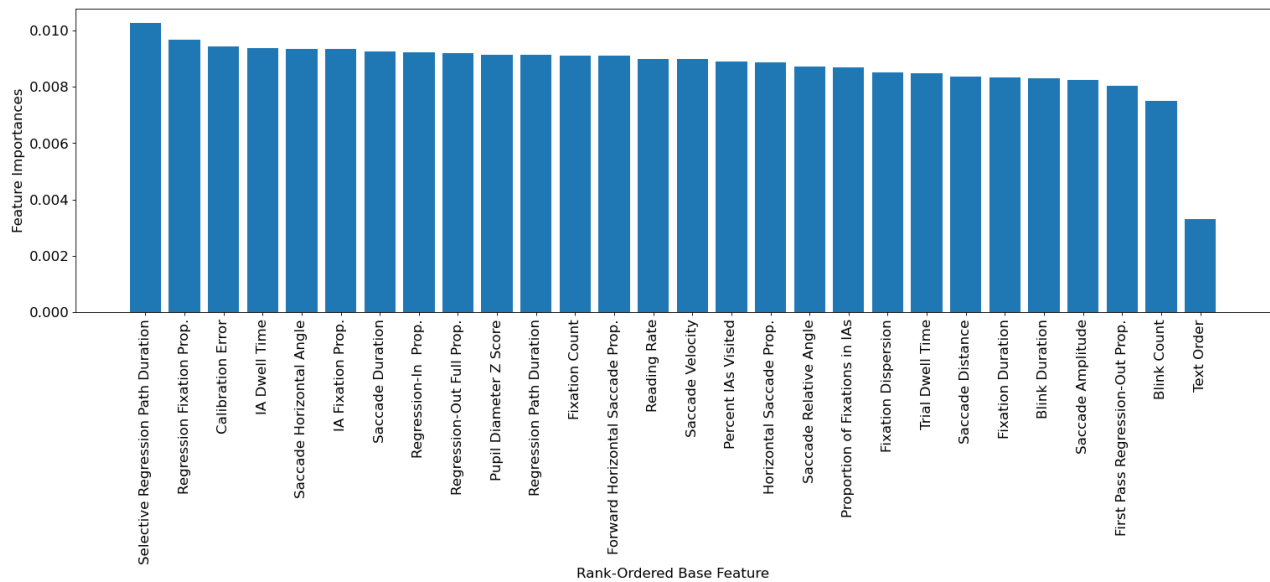


Figure 7. Rank-ordered mean feature importance across assessments.

Table 7. Pairwise feature importance Pearson correlations averaged over runs

Assessment	Rote: During	Rote: After Each	Rote: After All	Rote: Delay	Inference: During	Inference: After Each	Inference: After All	Inference: Delay
Rote: During	-	-0.01	0.16	0.06	-0.01	0.10	-0.10	0.02
Rote: After Each		-	0.27	0.30	0.35	0.10	0.32	-0.05
Rote: After All			-	0.35	0.06	0.35	0.05	-0.03
Rote: Delay				-	0.57	0.26	0.46	-0.04
Inference: During					-	0.15	0.57	-0.14
Inference: After Each						-	0.13	-0.05
Inference: After All							-	-0.10
Inference: Delay								-

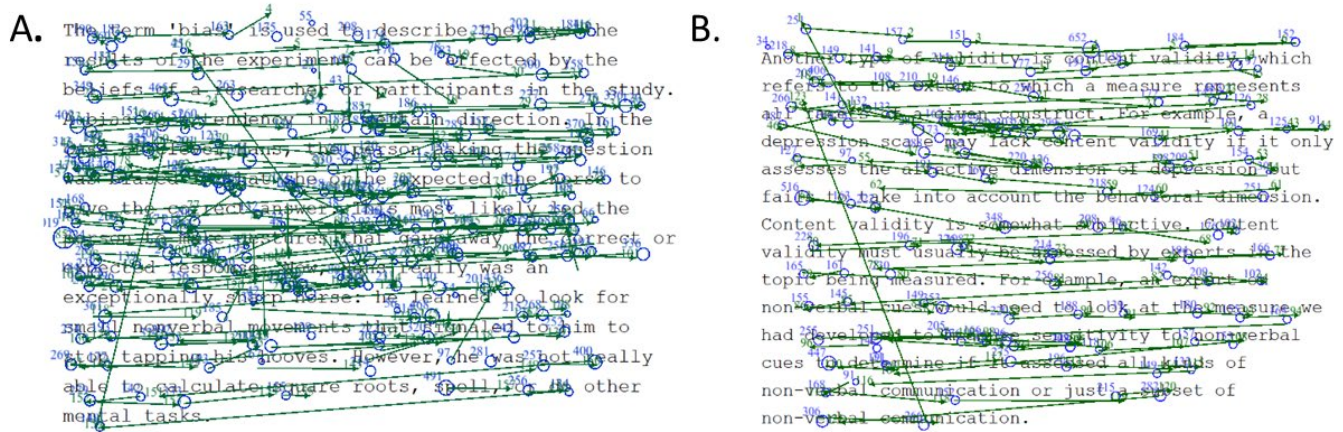


Figure 8. Example gaze on pages with accurate (A) and inaccurate (B) comprehension on a question asked immediately after reading the page. Saccades are shown with green lines and fixations with blue circles.

5. DISCUSSION

Our goal was to investigate the relationship between eye tracking and reading comprehension outcomes by identifying whether gaze-based models of reading could predict different depths of comprehension at varying degrees of delay. We discuss our main findings followed by applications, and limitations.

5.1 Main Findings

Beginning with *observed* rote and inference comprehension, we found that accuracies were highest during reading and dropped when assessed at seven days post-study, which is what we would expect. Nevertheless, the gaze-based Random Forest models could still predict both comprehension types significantly better than baselines during all stages of reading except for the “after each” assessment, which followed the same trend. Furthermore, we found a pattern such that gaze became more predictive of comprehension as time progressed within the main reading session, with performance peaking for assessments administered after reading all the texts.

As noted in Figure 1, theories of reading comprehension posit that readers first attend to and encode the text at the surface-level to form a text-based representation followed by a situational (mental) model via bridging inferences and elaboration, which is then consolidated into long-term memory. While there are reasons to expect that gaze should predict rote, short-term retention of information that simply requires attention to and basic encoding of the text (as others have shown – see Table 1), our results indicate that it is also related to the processes underlying deep, long-term comprehension which are typically viewed as internal cognitive processes that occur in the absence of eye movements [40]. To our knowledge, this is the first study to show that gaze during reading is predictive of deep, long-term comprehension since previous work has mainly focused on rote comprehension [55].

Second, while our models demonstrated participant-level generalizability (in contrast to prior work – see Table 1), the percent improvements of our models relative to baseline were admittedly modest. However, as indicated in the Introduction, the present goal was not to develop the most predictive model but to examine the predictive accuracy of a set of eye gaze features across comprehension depth and time. As such, we did not include many pertinent non-gaze features (e.g., text difficulty; item difficulty) that may be highly predictive when coupled with eye gaze.

It should be noted that D’Mello et al. [16] found that a gaze model yielded very high accuracies (AUROCs close to 0.9) for modeling rote comprehension during reading. One key difference is the D’Mello et al. study only measured rote comprehension with the items triggered in response to an automated mind-wandering detector. This may have engendered a very different reading strategy (and corresponding eye movement patterns) entailing skimming the text to identify targets for subsequent questions (the authors acknowledge this in Mills et al. [14]) than the more general reading strategies required here given the variety of comprehension measures. Thus, the high-accuracy scores reported in D’Mello et al. [16] might not generalize more broadly.

Third, there was overlap in feature importances across depth and time, with the top four predictive features being the selective regression path duration, regression fixation proportion, calibration error, and dwell time per interest area. It is possible these features capture measures of processing later in the time course of reading, and rereading text for comprehension repair, both of which are key to higher-level comprehension [9]. Dwell time, for instance, reflects the processing time (early and late), and in combination with selective regression path duration could be indicating processing difficulty later in reading a page (e.g., low selective regression path duration but high dwell time suggests difficulties only later on in the comprehension of a page [36]). Regression fixation proportion is also an important indication of comprehension repair [46, 52], and has been used in previous gaze models of comprehension [55]. Because it is difficult to interpret the direction of association of individual features in random forest models (due to interactivity), these patterns are speculative, and await further empirical data.

Historically, regressions have been one of the more difficult aspects of eye behavior to capture [48]. While it has long been posited that regressions occur when a reader experiences a difficulty in comprehension which triggers the reader to look back in the text to repair their comprehension deficit [9], several studies have linked an increase in regressions to better comprehension [28, 52], while others show the opposite effect [31]. Regression fixation proportion, the feature with the second highest average importance, indicates how often readers did not understand the text and acted to repair their comprehension [46], and might also be a way to distinguish better from poor readers, as better readers reread less and are more adept at redirecting their gaze efficiently [64]. Figure 8 shows an example of gaze behavior that leads to accurate and inaccurate comprehension. Note that in Figure 8A there are more regressions (seen as

long saccades cutting across multiple lines of text) and more reading in the middle of the text- an area which might have been giving the student some difficulty. On the other hand, the gaze on the page with inaccurate comprehension demonstrates a more even pattern of eye movements: possibly less attention to the text and less comprehension repair.

We also found that calibration error was predictive of comprehension. Indeed, prior work has found that greater pre-trial fixation dispersion is predictive of mind wandering [65]. Because mind wandering is negatively related to comprehension [14, 54], it might be the case that calibration error (and pre-trial fixation dispersion) is also negatively predictive of comprehension.

5.2 Applications

This research is a step towards gaze-based computational models of reading comprehension. Such models can be integrated into adaptive systems that trigger assessments and provide opportunities to correct comprehension deficiencies when lapses of comprehension are detected (similar to the gaze-based models that adaptively trigger interventions when mind wandering is detected [4, 13, 19, 27]). Given the modest accuracies obtained in the present study, the most immediate applications are in interventions that can be applied in a ‘fail-soft’ manner. These do not disrupt the student and do not pose any harm if comprehension is miss-classified. Interleaving questions during reading is one such example [57], as is encouraging re-reading at the end of a text or adaptively selecting post-reading assessments based on model-assessed comprehension during reading.

With further research, more fine-grained interventions that target different depths (rote vs. inference) and timescales (short- or long-term comprehension) are also feasible. For instance, if a student is preparing for an upcoming examination, models and interventions supporting long-term comprehension might be preferred compared to cases where short-term retention suffices (reading a short article). Other possible interventions include reducing textual difficulty or providing scaffolds when comprehension difficulties are detected [53] or even increasing difficulty when the reader is not being sufficiently challenged (e.g., the reverse cohesion effect [44] where good comprehenders benefit more from texts with lower cohesion).

In addition to direct intervention, the models also have applications with respect to assessment. For example, if the rote and inference models consistently (i.e., across multiple participants) predict high and low comprehension scores on a given page, respectively, this might suggest that there is a cohesion gap with respect to the content on the page that is impeding inference generation.

5.3 Limitations

Like all studies, ours has limitations. First, we only examined gaze on a particular page, thereby overly constraining the models. Therefore, there may have been other factors, such as gaze on the preceding page, that might have been relevant to reading comprehension but were not incorporated into the gaze models.

Furthermore, it is possible that the lab settings changed behaviors relative to how participants would behave in more ecologically valid settings. Specifically, participants donned other sensors and face masks to adhere to COVID safety procedures, which may have resulted in discomfort and unnatural reading behaviors (but see Table 4 which showed high horizontal saccade proportion and percent of fixations in interest areas indicating people were on task).

Next, the classes in the data were imbalanced, and we chose to not balance the classes since this might not capture real-world variation in comprehension. However, class imbalance might have introduced a confound when comparing model performance over time in that accuracies reflected the level of class imbalance rather than differences in comprehension depth and time. To address this possibility, we did test models on balanced classes and results did not change.

Although the present study demonstrated cross-participant generalization, participants only read one set of texts and therefore it is unknown whether the models would generalize to new texts. That said, because we used features which capture relative changes in gaze (e.g., angles) as opposed to absolute, stimulus-dependent values (e.g., coordinates), we think they are likely to generalize to similar contexts. To this point, prior work using similar global page-level features demonstrated cross-task-generalization for mostly rote comprehension after reading [55], but this is an empirical question for comprehension models at different depths and time delays.

Finally, the LSTMs yielded chance-level performance. This was despite using features that reflected relative changes in gaze (e.g., relative angles) in contrast to prior LSTM work that used absolute fixation coordinates [1]. It might be the case that there are not generalizable patterns in local gaze dynamics that are predictive of comprehension. Alternatively, and more likely, there might not have been sufficient data to learn these patterns should they exist given the relatively small number of training examples compared to the number of parameters in the LSTM models.

5.4 Concluding Remarks

Reading comprehension is a complex cognitive process that is critical to daily tasks. It unfolds across different depths and over time, raising the question of what eye movements known to index initial encoding of information can reveal about the processes underlying deep, long-term comprehension (Figure 1). Our results show, for the first time, that eye movements have the potential to provide an index into deeper inference-level comprehension assessed as late as a week after reading, indicating they capture far more than temporary surface-level encoding of a text.

6. ACKNOWLEDGMENTS

This research was supported by the National Science Foundation (DRL 1920510). The opinions expressed are those of the authors and do not represent views of the funding agencies.

7. REFERENCES

- [1] Ahn, S., Kelton, C., Balasubramanian, A. and Zelinsky, G. 2020. Towards Predicting Reading Comprehension From Gaze Behavior. *ACM Symposium on Eye Tracking Research and Applications*. (2020), 1–5. DOI:<https://doi.org/10.1145/3379156.3391335>.
- [2] Bates, D., Mächler, M., Bolker, B. and Walker, S. 2015. Fitting Linear Mixed-Effects Models Using lme4. *Journal of Statistical Software*. 67, (Oct. 2015), 1–48. DOI:<https://doi.org/10.18637/jss.v067.i01>.
- [3] Biedert, R., Buscher, G., Schwarz, S., Möller, M., Dengel, A. and Lottermann, T. 2010. The text 2.0 framework: writing web-based gaze-controlled realtime applications quickly and easily. *Proceedings of the 2010 workshop on Eye gaze in intelligent human machine interaction - EGIHMI '10*. (2010), 114–117. DOI:<https://doi.org/10.1145/2002333.2002351>.

- [4] Bixler, R. and D’Mello, S. 2016. Automatic gaze-based user-independent detection of mind wandering during computerized reading. *User Modeling and User-Adapted Interaction*. 26, 1 (Mar. 2016), 33–68. DOI:https://doi.org/10.1007/s11257-015-9167-1.
- [5] Booth, R.W. and Weger, U.W. 2013. The function of regressions in reading: Backward eye movements allow rereading. *Memory & Cognition*. 41, 1 (Jan. 2013), 82–97. DOI:https://doi.org/10.3758/s13421-012-0244-y.
- [6] Chi, M.T. 2000. Self-explaining expository texts: The dual processes of generating inferences and repairing mental models. *Advances in instructional psychology*. 5, (2000), 161–238.
- [7] Christianson, K., Luke, S.G., Hussey, E.K. and Wochna, K.L. 2017. Why reread? Evidence from garden-path and local coherence structures. *Quarterly Journal of Experimental Psychology*. 70, 7 (Jul. 2017), 1380–1405. DOI:https://doi.org/10.1080/17470218.2016.1186200.
- [8] Conati, C., Aleven, V., Mitrovic, A., Graesser, A., Sottolare, A., Hu, H. and Holden, H. 2013. Eye-Tracking for Student Modelling in Intelligent Tutoring Systems. *Design Recommendations for Intelligent Tutoring Systems - Volume 1: Learner Modeling*. Army Research Laboratory. 227–236.
- [9] Cook, A.E. and Wei, W. 2019. What Can Eye Movements Tell Us about Higher Level Comprehension? *Vision*. 3, 3 (Sep. 2019), 45. DOI:https://doi.org/10.3390/vision3030045.
- [10] Copeland, L. and Gedeon, T. 2013. Measuring reading comprehension using eye movements. *2013 IEEE 4th International Conference on Cognitive Infocommunications (CogInfoCom)* (Dec. 2013), 791–796.
- [11] Copeland, L., Gedeon, T. and Caldwell, S. 2016. Effects of text difficulty and readers on predicting reading comprehension from eye movements. *2015 6th IEEE International Conference on Cognitive Infocommunications (CogInfoCom)*. (Jan. 2016), 407–412. DOI:https://doi.org/10.1109/coginfocom.2015.7390628.
- [12] Copeland, L., Gedeon, T. and Mendis, S. 2014. Fuzzy Output Error as the Performance Function for Training Artificial Neural Networks to Predict Reading Comprehension from Eye Gaze. (2014), 586–593. DOI:https://doi.org/10.1007/978-3-319-12637-1_73.
- [13] D’Mello, S., Kopp, K., Bixler, R.E. and Bosch, N. 2016. Attending to Attention: Detecting and Combating Mind Wandering during Computerized Reading. *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems - CHI EA ’16* (San Jose, California, USA, 2016), 1661–1669.
- [14] D’Mello, S. and Mills, C. Mind wandering during reading: An interdisciplinary and integrative review of psychological, computing, & intervention research and theory. *Language and Linguistics Compass: Cognitive Science of Language*.
- [15] D’Mello, S.K. 2016. Giving Eyesight to the Blind: Towards Attention-Aware AIED. *International Journal of Artificial Intelligence in Education*. 26, 2 (Jun. 2016), 645–659. DOI:https://doi.org/10.1007/s40593-016-0104-1.
- [16] D’Mello, S.K., Southwell, R. and Gregg, J. 2020. Machine-Learned Computational Models Can Enhance the Study of Text and Discourse: A Case Study Using Eye Tracking to Model Reading Comprehension. *Discourse Processes*. 57, 5–6 (Jul. 2020), 420–440. DOI:https://doi.org/10.1080/0163853X.2020.1739600.
- [17] D’Mello, S.K., Tay, L. and Southwell, R. 2022. Psychological Measurement in the Information Age: Machine-Learned Computational Models. *Current Directions in Psychological Science*. 31, 1 (Feb. 2022), 76–87. DOI:https://doi.org/10.1177/09637214211056906.
- [18] Engbert, R., Nuthmann, A., Richter, E.M. and Kliegl, R. 2005. SWIFT: A Dynamical Model of Saccade Generation During Reading. *Psychological Review*. 112, 4 (2005), 777–813. DOI:https://doi.org/10.1037/0033-295X.112.4.777.
- [19] Faber, M., Bixler, R. and D’Mello, S.K. 2018. An automated behavioral measure of mind wandering during computerized reading. *Behavior Research Methods*. 50, 1 (Feb. 2018), 134–150. DOI:https://doi.org/10.3758/s13428-017-0857-y.
- [20] Faber, M., Krasich, K., Bixler, R.E., Brockmole, J.R. and D’Mello, S.K. 2020. The eye–mind wandering link: Identifying gaze indices of mind wandering across tasks. *Journal of Experimental Psychology: Human Perception and Performance*. 46, 10 (Oct. 2020), 1201–1221. DOI:https://doi.org/10.1037/xhp0000743.
- [21] Faraway, J.J. 2016. *Extending the Linear Model with R: Generalized Linear, Mixed Effects and Nonparametric Regression Models, Second Edition*. CRC Press.
- [22] Flesch, R. 1948. A new readability yardstick. *Journal of applied psychology*. 32, 3 (1948), 221.
- [23] Frazier, L. and Rayner, K. 1982. Making and correcting errors during sentence comprehension: Eye movements in the analysis of structurally ambiguous sentences. *Cognitive Psychology*. 14, 2 (Apr. 1982), 178–210. DOI:https://doi.org/10.1016/0010-0285(82)90008-1.
- [24] Graesser, A.C., Singer, M. and Trabasso, T. 1994. Constructing inferences during narrative text comprehension. *Psychological Review*. 101, 3 (1994), 371–395. DOI:https://doi.org/10.1037/0033-295X.101.3.371.
- [25] Grainger, J. and Holcomb, P.J. 2009. Watching the Word Go by: On the Time-course of Component Processes in Visual Word Recognition. *Language and Linguistics Compass*. 3, 1 (2009), 128–156. DOI:https://doi.org/10.1111/j.1749-818x.2008.00121.x.
- [26] Henderson, J.M., Choi, W., Luke, S.G. and Schmidt, J. 2018. Neural correlates of individual differences in fixation duration during natural reading. *Quarterly Journal of Experimental Psychology*. 71, 1 (Jan. 2018), 314–323. DOI:https://doi.org/10.1080/17470218.2017.1329322.
- [27] Hutt, S., Krasich, K., Mills, C., Bosch, N., White, S., Brockmole, J.R. and D’Mello, S.K. 2019. Automated gaze-based mind wandering detection during computerized learning in classrooms. *User Modeling and User-Adapted Interaction*. 29, 4 (Sep. 2019), 821–867. DOI:https://doi.org/10.1007/s11257-019-09228-5.
- [28] Inhoff, A.W., Kim, A. and Radach, R. 2019. Regressions during Reading. *Vision*. 3, 3 (2019), 35. DOI:https://doi.org/10.3390/vision3030035.
- [29] Jeni, L.A., Cohn, J.F. and De La Torre, F. 2013. Facing Imbalanced Data—Recommendations for the Use of Performance Metrics. *2013 Humaine Association Conference on Affective Computing and Intelligent Interaction* (Sep. 2013), 245–251.
- [30] Just, M.A. and Carpenter, P.A. 1980. A theory of reading: From eye fixations to comprehension. *Psychological Review*. 87, 4 (Jul. 1980), 329–354. DOI:http://dx.doi.org/10.1037/0033-295X.87.4.329.
- [31] Kemper, S., Crow, A. and Kemtes, K. 2004. Eye-Fixation Patterns of High- and Low-Span Young and Older Adults: Down the Garden Path and Back Again. *Psychology and Aging*. 19, 1 (2004), 157–170. DOI:https://doi.org/10.1037/0882-7974.19.1.157.

- [32] Kendeou, P. and van den Broek, P. 2007. The effects of prior knowledge and text structure on comprehension processes during reading of scientific texts. *Memory & Cognition*. 35, 7 (Oct. 2007), 1567–1577. DOI:<https://doi.org/10.3758/BF03193491>.
- [33] Kintsch, W. 2011. An Overview of Top-Down and Bottom-Up Effects in Comprehension: The CI Perspective. *Discourse Processes*. 39, 2–3 (2011), 125–128. DOI:<https://doi.org/10.1080/0163853x.2005.9651676>.
- [34] Kintsch, W. 2005. An Overview of Top-Down and Bottom-Up Effects in Comprehension: The CI Perspective. *Discourse Processes*. 39, 2–3 (May 2005), 125–128. DOI:<https://doi.org/10.1080/0163853X.2005.9651676>.
- [35] Kruthiventi, S.S.S., Ayush, K. and Babu, R.V. 2017. DeepFix: A Fully Convolutional Neural Network for Predicting Human Eye Fixations. *IEEE Transactions on Image Processing*. 26, 9 (Sep. 2017), 4446–4456. DOI:<https://doi.org/10.1109/TIP.2017.2710620>.
- [36] Liversedge, S.P., Paterson, K.B. and Pickering, M.J. 1998. Eye Movements and Measures of Reading Time. *Eye Guidance in Reading and Scene Perception*. Elsevier. 55–75.
- [37] Masson, M.E.J. 1983. Conceptual processing of text during skimming and rapid sequential reading. *Memory & Cognition*. 11, 3 (1983), 262–274. DOI:<https://doi.org/10.3758/bf03196973>.
- [38] McDonald, S.A. and Shillcock, R.C. 2003. Eye movements reveal the on-line computation of lexical probabilities during reading. *Psychological Science*. 14, 6 (Nov. 2003), 648–652. DOI:https://doi.org/10.1046/j.0956-7976.2003.psci_1480.x.
- [39] McNamara, D.S. and Magliano, J. 2009. Chapter 9 Toward a Comprehensive Model of Comprehension. *Psychology of Learning and Motivation*. Academic Press. 297–384.
- [40] Meseguer, E., Carreiras, M. and Clifton, C. 2002. Overt reanalysis strategies and eye movements during the reading of mild garden path sentences. *Memory & Cognition*. 30, 4 (Jun. 2002), 551–561. DOI:<https://doi.org/10.3758/BF03194956>.
- [41] Mills, C., Gregg, J., Bixler, R. and D’Mello, S.K. 2021. Eye-Mind reader: an intelligent reading interface that promotes long-term comprehension by detecting and responding to mind wandering. *Human-Computer Interaction*. 36, 4 (Jul. 2021), 306–332. DOI:<https://doi.org/10.1080/07370024.2020.1716762>.
- [42] Miyata, H., Minagawa-Kawai, Y., Watanabe, S., Sasaki, T. and Ueda, K. 2012. Reading Speed, Comprehension and Eye Movements While Reading Japanese Novels: Evidence from Untrained Readers and Cases of Speed-Reading Trainees. *PLoS ONE*. 7, 5 (May 2012). DOI:<https://doi.org/10.1371/journal.pone.0036091>.
- [43] Nadel, L., Hupbach, A., Gomez, R. and Newman-Smith, K. 2012. Memory formation, consolidation and transformation. *Neuroscience & Biobehavioral Reviews*. 36, 7 (2012), 1640–1645. DOI:<https://doi.org/10.1016/j.neubiorev.2012.03.001>.
- [44] O’reilly, T. and McNamara, D.S. 2007. Reversing the Reverse Cohesion Effect: Good Texts Can Be Better for Strategic, High-Knowledge Readers. *Discourse Processes*. 43, 2 (Apr. 2007), 121–152. DOI:<https://doi.org/10.1080/01638530709336895>.
- [45] Price, C.J., Moore, C.J., Humphreys, G.W. and Wise, R.J.S. 1997. Segregating Semantic from Phonological Processes during Reading. *Journal of Cognitive Neuroscience*. 9, 6 (1997), 727–733. DOI:<https://doi.org/10.1162/jocn.1997.9.6.727>.
- [46] Rayner, K. 1998. Eye movements in reading and information processing: 20 years of research. *Psychological Bulletin*. 124, 3 (Nov. 1998), 372–422. DOI:<https://doi.org/10.1037/0033-2909.124.3.372>.
- [47] Rayner, K. and McConkie, G.W. 1976. What guides a reader’s eye movements? *Vision Research*. 16, 8 (Jan. 1976), 829–837. DOI:[https://doi.org/10.1016/0042-6989\(76\)90143-7](https://doi.org/10.1016/0042-6989(76)90143-7).
- [48] Rayner, K. and Slattery, T.J. 2009. Eye movements and moment-to-moment comprehension processes in reading. *Beyond decoding: The behavioral and biological foundations of reading comprehension*. Guilford Press. 27–45.
- [49] Rayner, K., Slattery, T.J., Drieghe, D. and Liversedge, S.P. 2011. Eye Movements and Word Skipping During Reading: Effects of Word Length and Predictability. *Journal of Experimental Psychology: Human Perception and Performance*. 37, 2 (2011), 514–528. DOI:<https://doi.org/10.1037/a0020990>.
- [50] Reichle, E.D., Warren, T. and McConnell, K. 2009. Using E-Z reader to model the effects of higher level language processing on eye movements during reading. *Psychonomic Bulletin & Review*. 16, 1 (Feb. 2009), 1–21. DOI:<https://doi.org/10.3758/PBR.16.1.1>.
- [51] Roda, C. and Thomas, J. 2006. Attention aware systems: Theories, applications, and research agenda. *Computers in Human Behavior*. 22, 4 (Jul. 2006), 557–587. DOI:<https://doi.org/10.1016/j.chb.2005.12.005>.
- [52] Schotter, E.R., Tran, R. and Rayner, K. 2014. Don’t Believe What You Read (Only Once): Comprehension Is Supported by Regressions During Reading. *Psychological Science*. 25, 6 (Jun. 2014), 1218–1226. DOI:<https://doi.org/10.1177/0956797614531148>.
- [53] Shute, V.J. and Zapata-Rivera, D. 2012. Adaptive Technologies for Training and Education. (2012), 7–27. DOI:<https://doi.org/10.1017/cbo9781139049580.004>.
- [54] Smallwood, J., McSpadden, M. and Schooler, J.W. 2008. When attention matters: the curious incident of the wandering mind. *Memory & cognition*. 36, 6 (2008), 1144–50. DOI:<https://doi.org/10.3758/mc.36.6.1144>.
- [55] Southwell, R., Gregg, J., Bixler, R. and D’Mello, S.K. 2020. What Eye Movements Reveal About Later Comprehension of Long Connected Texts. *Cognitive Science*. 44, 10 (2020), e12905. DOI:<https://doi.org/10.1111/cogs.12905>.
- [56] Strukelj, A. and Niehorster, D.C. 2018. One page of text: Eye movements during regular and thorough reading, skimming, and spell checking. *Journal of Eye Movement Research*. 11, 1 (Feb. 2018). DOI:<https://doi.org/10.16910/jemr.11.1.1>.
- [57] Szpunar, K.K., Khan, N.Y. and Schacter, D.L. 2013. Interpolated memory tests reduce mind wandering and improve learning of online lectures. *Proceedings of the National Academy of Sciences*. 110, 16 (Apr. 2013), 6313–6317. DOI:<https://doi.org/10.1073/pnas.1221764110>.
- [58] Turan, C., Wang, Y., Lai, S.-C., Neergaard, K.D. and Lam, K.-M. 2018. Facial Expressions of Sentence Comprehension. *2018 IEEE 23rd International Conference on Digital Signal Processing (DSP)* (Nov. 2018), 1–5.
- [59] Van Dijk, T.A. and Kintsch, W. 1983. Strategies of discourse comprehension. (1983).
- [60] Veldre, A. and Andrews, S. 2014. Lexical Quality and Eye Movements: Individual Differences in the Perceptual Span

- of Skilled Adult Readers. *Quarterly Journal of Experimental Psychology*. 67, 4 (Apr. 2014), 703–727. DOI:<https://doi.org/10.1080/17470218.2013.826258>.
- [61] Vrzakova, H., Amon, M.J., Stewart, A.E.B. and D’Mello, S.K. 2019. Dynamics of Visual Attention in Multiparty Collaborative Problem Solving using Multidimensional Recurrence Quantification Analysis. *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, May 2019), 1–14.
- [62] Wang, X., Zhao, X. and Ren, J. 2019. A New Type of Eye Movement Model Based on Recurrent Neural Networks for Simulating the Gaze Behavior of Human Reading. *Complexity*. 2019, (2019), 1–12. DOI:<https://doi.org/10.1155/2019/8641074>.
- [63] Yeari, M. 2017. The role of working memory in inference generation during reading comprehension_ Retention, (re)activation, or suppression of verbal information? *Learning and Individual Differences*. 56, (2017), 1–12. DOI:<https://doi.org/10.1016/j.lindif.2017.04.002>.
- [64] Zabrocky, K. and Commander, N.E. 1993. Rereading to Understand: The Role of Text Coherence and Reader Proficiency. *Contemporary Educational Psychology*. 18, 4 (Oct. 1993), 442–454. DOI:<https://doi.org/10.1006/ceps.1993.1033>.
- [65] Zeng, H. and Jonides, J. 2021. Pre-trial Gaze Stability Predicts Momentary Slips of Attention. *PsyArXiv*. (2021).

Predicting Reading Comprehension Scores of Elementary School Students

Bruce (Yuyang) Nie¹

Hélène Deacon²

Alona Fyshe³

Carrie Demmans Epp¹

¹ EdTeKLA Group, University of Alberta, {ynie, cdemmansepp}@ualberta.ca

² Dalhousie University, Helene.Deacon@dal.ca

³ University of Alberta, Alberta Machine Intelligence Institute, alona@ualberta.ca

ABSTRACT

A child's ability to understand text (reading comprehension) can greatly impact both their ability to learn in the classroom and their future contributions to society. Reading comprehension draws on oral language; behavioural measures of knowledge at the word and sentence levels have been shown to be related to children's reading comprehension. In this study, we examined the impact of word and sentence level text-features on children's reading comprehension. We built a predictive model that uses natural language processing techniques to predict the question-level performance of students on reading comprehension tests. We showed that, compared to a model that used measures of student knowledge and subskills alone, a model that used features of sentence complexity, lexical surprisal, rare word use, and general context improved prediction accuracy by more than four percentage points. Our subsequent analyses revealed that these features compensate for the shortcomings of each other and work together to produce maximal performance. This provides insight into how different characteristics of the text and questions can be used to predict student performance, leading to new ideas about how text and reading comprehension interact. Our work also suggests that using a combination of text features could support the adaptation of reading materials to meet student needs.

Keywords

Reading comprehension, text-features, early education, natural language processing, learner modelling

1. INTRODUCTION

Elementary students' ability to understand text, or their reading comprehension (e.g., [30]), supports their successful participation in both education and society [7]. It is well-established that reading comprehension is determined, at least in part, by children's skill in oral language [19], with strong impacts of oral language skills—at both the word and sentence levels (i.e., vocabulary and syntax)—on reading comprehension skills (e.g., [11]). We built on these ideas and tested whether automatically extracted indicators of vocabulary and syntax complexity would be indicative of reading

comprehension scores. To address this question, we extract features from a widely used standardised measure of reading comprehension [34]. Thus, we introduce various text-based measures that can be automatically extracted. After extracting these features, we trained and evaluated models. We then tested their relative performance and analyzed the added value of including all of these features in the context of a predictive neural network model. Our modelling approach could also be used to analyze new reading comprehension tests or tasks, enabling others to better understand how student knowledge and subskills interact with the characteristics of the text to influence reading comprehension.

The educational value of this work lies in informing the optimal selection of texts and questions to support the development of children's reading comprehension. Predictions on where mistakes would likely be made by individual students could be used to support downstream tasks, such as adapting systems so that both the comprehension questions asked and the assigned texts challenge students appropriately [57]. Further, interpreting the predictions made by such models and the model attributes may improve our understanding of what contributes to reading comprehension challenges, which could support the design of optimally effective interventions.

2. RELATED WORK

The single most widely cited theory of reading comprehension is the Simple View of Reading [19]. According to this theory, children's reading comprehension is determined by the product of their word reading and their oral language skills. Behavioural research since its development has made good progress in identifying the individual oral language skills that support reading comprehension. Two of the best-established lie in vocabulary, or knowledge of individual word meanings, and syntactic awareness, or the ability to reflect on or manipulate spoken sentences [5]. Each of vocabulary and syntactic awareness are strong predictors of children's levels of reading comprehension [54]. For instance, Deacon and Kieffer [11] showed that children's ability to manipulate sentences in Grade 3 predicted gains made in reading comprehension between Grades 3 and 4, a contribution similar in magnitude to that of word-level reading. There is an even larger body of research demonstrating the impacts of children's vocabulary knowledge on their ability to understand what they read [43]. Together, this body of work shows that individual differences in oral language skills—at both the word and sentence levels—are related to children's ability to understand the texts that they read.

But what about the features of the texts themselves?

Y. Nie, H. Deacon, A. Fyshe, and C. D. Epp. Predicting reading comprehension scores of elementary school students. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 158–170, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852952>

It is well-established that the features of a text impact a person's comprehension of that text. These features can be identified manually, or we can use techniques from natural language processing (NLP) to identify the qualities of a text and extract their associated features. The field of NLP has developed many approaches that can be used to identify syntax, semantics, and word use. There are also methods that can identify the broader relationships in people's language use once those methods have been trained with corpora.

Of the techniques that do not require training with a corpus, are those that predominantly count elements of a text to derive measures that can then be used as features in a model. One widely-used tool that employs this approach to capture the syntactic complexity of sentences is Coh-Metrix. Coh-Metrix tracks the linguistic properties of a text using features such as the average length of words in a sentence or number of sentences in a paragraph [23]. The most recent version of Coh-Metrix also incorporates the scoring of connections between sentences, such as noun overlap between adjacent sentences and causal cohesion (sentences that are linked by causal connectives) [22].

Given the importance of syntactic complexity to reading comprehension, it is not surprising that Coh-Metrix has been used to extract the linguistic properties of texts to predict student reading comprehension scores in the context of the iStart tutoring system [1]. iStart provided training in active reading strategies (i.e., self-explanation) to support improved reading comprehension. In this case, features were extracted from student self-explanations and used to predict reading comprehension scores on the Gates MacGinitie test. These analyses provided insight into how the language produced by students can be used to predict their overall reading comprehension performance. Including the simple linguistic features that were extracted using Coh-Metrix accounted for additional variance in student performance, which demonstrated that the inclusion of such features has the potential to support learner modelling tasks.

Building on the idea of Coh-Metrix, AMOC aimed to automatically capture the semantic complexity and context of a text [8]. It uses similar approaches to those embedded within Coh-Metrix and adds semantic links using dictionary-like tools and Word2Vec. These links are then used to output a graph representation of the semantic model (context) associated with a segment of text.

Language models are a potential alternative approach to automatically capturing the syntactic and semantic or contextual features of a text. Existing language modelling approaches vary in complexity from relatively simple probabilistic representations of language (e.g., n-grams) to more complex neural network-based approaches (e.g., RNNs). All of these language modelling approaches require a corpus of data from which they can learn language-usage patterns.

Assuming you have access to a corpus, n-grams can be used to capture both the syntactic and semantic constraints in a text by determining the probability with which words are expected to follow one another. Mitchell et al [39] investigated using trigrams (i.e., n-grams of length 3) to analyze reading times and understand reading comprehension. They used these n-grams to estimate the lexical surprisal associated with a text and found a correlation between the surprisal measure and reading time. Similarly, a study by Hofmann et al [29] used a trigram language model to predict reading speed from eye-tracking data while acknowledging that trigrams do not account for long-range semantic effects (unlike AMOC and some more advanced language modelling approaches). The finding that reading time may be related to comprehension is not directly

measurable with n-grams; rather, they enable the extraction of measures, such as lexical surprisal, that can be used to predict reading time.

To better model long-range semantics and sentence meaning, recurrent neural networks (RNN) have been used to improve predictions of reading comprehension scores and reading times. Frank and Hoeks [18] used a specific type of RNN, called long short-term memory (LSTM), to correlate reading times with comprehension. Through this investigation, they found RNNs can capture both the structure and semantics of a long text thus improving reading time predictions. Transformer-based neural network models [53], such as BERT [15], have shown superior performance over RNNs when conducting various NLP tasks, such as neural machine translation [31] and sentiment analysis [38]. This suggests the use of transformer-based models may support improved performance when predicting reading comprehension scores.

Outside of the use of language models, other approaches to representing student performance (learner or student models) have been attempted in systems that rely on language as a primary source of interaction with the student. One project aimed to predict student scores on a post-test using the logs of student interactions from intelligent tutoring systems (ITS) that teach physics and probability using word-based problems [35]. Mao et al. [35] collected data from two ITSs and used the training content to predict student post-test results. They trained a Bayesian knowledge tracing (BKT) model to infer whether students had learned the intended units of knowledge (knowledge components). Their results showed BKT's prediction is more accurate than that of the LSTM on its own. A model that jointly used BKT and LSTM to predict post-test scores outperformed both individual models, suggesting the potential for combining different approaches to capturing student capabilities in relation to the complexities of the learning and assessment materials. This initial work has provided promising results by incorporating representations of the text as part of the prediction of student performance. However, the skill representations used by BKT models typically require the expert labelling of skills along with the careful development and sequencing of questions, and this sequencing is often specific to an ITS. Consequently, it cannot be easily generalized or applied to other prediction tasks.

Moreover, these types of approaches have yet to be applied to modelling student reading comprehension within e-learning systems. Within the instructional domain of literacy, previous attempts at modelling student skills or knowledge have focused on understanding student reading strategies [24], improving their vocabulary acquisition [14, 17], or improving reading proficiency (decoding and fluency) [2, 40], as steps towards supporting literacy. In these cases, the logs of student activities, their interactions with the system, and the texts they produce are used to model student knowledge and skills. These types of models are then used to provide feedback to students [13], inform teachers of student activities and skills, or adapt learning content [12, 13]. Consistent with the above modelling goals, the training provided by the e-learning systems that aim to support student literacy has tended to focus on exposure-based approaches [3, 12], stealth assessment during game play [17], and the training or modelling of reading strategies [17, 21]. In the research on the use of software to support reading proficiency, tests of reading comprehension have been used to evaluate the effectiveness of the system [1, 16, 40] rather than being an integral part of the content that is adapted.

We do not yet have a model that can predict student performance on comprehension questions by capturing students' underlying oral language skills while accounting for text features. A model that

could do this would support the selection of both texts and questions within an adaptive learning environment like an ITS. The present study is a first attempt at developing such a model. It asks, ‘What is the added value of including text features as input when predicting the correctness of question answering in reading tasks?’

3. METHODS

To answer the above question, we applied supervised learning to archival data from a study of children’s reading development. We compared models that include different features we extracted from a reading comprehension test to a baseline that only included measures of student oral language skills and language knowledge.

3.1 Participants

This study was approved by the Social Sciences and Humanities Research Ethics Board of Dalhousie University. All children participated with parental consent and child assent. We recruited Grade 3 students from a combination of urban and rural public schools in Nova Scotia.

There were 139 students with a mean age of 8.80 years at the time of testing ($SD = 0.29$; range = 8.15–9.37 years). Of these students, 74 were boys and 65 were girls. Based on parent-report, most children were native speakers of English; 4 spoke a language other than English at home.

Table 1 shows the children’s performance on standardized measures (see section 2.3.2 for details on these measures). Student scores suggest that participants were typically developing.

Table 1. Descriptive Statistics for the Individual knowledge and sub-skill measure scores

Measure Name	Rel.	Mean	SD	Min	Max
PPVT-M	.84	31.95	4.93	19.00	46.00
CTOPP	.93	23.01	6.10	8.00	33.00
Syntax Awareness	.74	9.57	3.27	2.00	16.00
TOWRE	.93	56.97	14.40	4.00	87.00
WISC	.91	13.17	2.31	8.00	20.00
WRMT-3	.97	60.97	12.72	3.00	83.00

Note. Raw scores are reported for all measures. Reliability (Rel.) was retrieved from the instrument manual. SD - standard deviation.

3.2 Procedures

The measures reported on here were completed as a part of a larger battery, presented in two individual sessions and one group session (up to 12 children). We only report measures relevant to the present study.

In session 1, participants completed the Sight Word Efficiency subtest of the Test of Word Reading Efficiency (TOWRE), amongst other measures.

Session 2 was completed an average of two days after the first individual session ($M = 1.93$, $SD = 1.39$; range: 1 to 9 days). Participants completed the Word Identification subtest of the Woodcock Reading Mastery Tests (WRMT), a modified version of Peabody Picture Vocabulary Test (PPVT-M), Digit Span from the Wechsler Intelligence Scale for Children (WISC), and Elision from the Comprehensive Test of Phonological Processing (CTOPP).

The group session was completed an average of just over 2 days after the first individual session ($M = 2.27$, $SD = 2.12$; range: 1 to 9 days). In the group session, children completed the Comprehension subtest of the Gates–MacGinitie Reading Tests.

3.3 Measures

3.3.1 Reading Comprehension Measure

To measure reading comprehension, we administered the Level 3 Comprehension subtest of the fourth edition of the Gates–MacGinitie Reading Tests¹ [34] according to manual instructions. The manual reports a reliability of .93 for this instrument. Students were given 35 minutes to read 11 short texts and answer the three to six multiple choice questions following each text.

We aim to predict whether students correctly answered individual questions for each of the texts from this test.

3.3.2 Individual Knowledge and Sub-skill Measures

To incorporate differences in student oral language skills and knowledge, we administered tests to assess their vocabulary, phonological awareness, working memory, word reading fluency, and word reading accuracy. The descriptive statistics for these measures are shown in Table 1, and the instruments are detailed below.

Vocabulary knowledge. To measure receptive vocabulary knowledge, a shortened version (51 items) of the PPVT-3 [36] was used. For each item, students chose which of a set of four black-and-white pictures referred to an orally presented word. This shortened version (PPVT-M) has been validated with Grade 1 to 3 children [11].

Syntactic awareness. This was measured with an 18-item task in which children corrected sentences based on Deacon and Kieffer’s framework [11]. For instance, children are presented orally with the scrambled sentence, “From the library were stolen the books.” and they were asked to fix the sentence so that it sounds right (in this case, “the books were stolen from the library”). Children were given 3 practice items prior to completing the test.

Phonological awareness. We measured phonological awareness with the Elision subtest of the second edition of CTOPP [55]. Participants were asked to repeat words without pronouncing certain syllables or phonemes (e.g., *bold* without /b/). Phonological awareness was measured because of its association with word reading and reading comprehension [10].

Working memory. Given established correlations of working memory with both reading comprehension and word reading in 9-year-olds (e.g., [5]), we measured it using the Digit Span subtest of the fourth edition of the WISC [56]. In this task, participants repeat a series of digits of increasing length in the order given or the reverse order.

Word reading fluency. We measured word reading fluency using the Sight Word Efficiency subtest of the TOWRE [51]. Participants were given 45 seconds to read a list of words as fast as possible.

Word reading accuracy. We measured word reading accuracy with the Word Identification subtest of the WRMT-3 [58]. Students read words that became increasingly difficult.

¹ The test content is available at <https://edinstruments.com/instruments/gates-macginitie-reading-tests-4th-edition-gmrt-4>

3.3.3 Text-based Features

We extracted both word- and sentence-level text-based features. The features included sentence complexity, rare word use, lexical surprisal, and sentence context. The descriptive statistics of all extracted features can be seen in Table 2. Since rare word use represents a proportion of the text, it produces a single input feature. Similarly, question complexity produces a single input feature because questions are one sentence long. In contrast, lexical surprisal and sentence complexity for the text, which contains multiple sentences, are represented using three features: mean, standard deviation, and maximum.

3.3.3.1 Sentence Complexity

Complex sentences increase working memory load, which makes reading comprehension challenging [20]. We used the depth of the parse trees from the text as a proxy for sentence complexity. Each parse tree identifies the grammatical constituents of a sentence and communicates an aspect of a sentence’s complexity through its structure.

Table 2 Descriptive Statistics for the Text-based Features

Feature Name	Mean	SD	Min	Max
Sentence Complexity				
Mean	8.41	0.88	7.18	10.33
SD	2.32	0.74	1.34	3.96
Maximum	12.33	1.75	10.00	16.00
Lexical Surprisal				
Mean	.44	.30	0	1
SD	.49	.26	0	1
Maximum	.56	.30	0	1
Rare Word Use	.46	.06	.35	.56
Question Complexity	8.37	1.94	6	13

Note. *SD* - standard deviation.

To obtain this sentence-level measure, we first tagged the sentences from each text in the reading comprehension test with their associated parts of speech (POS). From the tagged words, we built parse trees using a probabilistic context-free grammar (PCFG). Both tagging and parsing were performed by the CoreNLP tagging tool [52].

Figure 1 shows a parse tree for the “Snow turns blue when blue ice-worms live in it.” This sentence is taken from one of the texts that was used to measure reading comprehension.

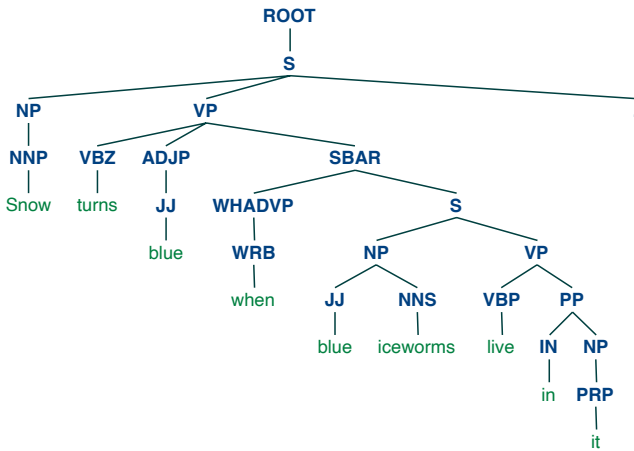


Figure 1. The parse tree with the highest probability for the sentence “Snow turns blue when blue ice-worms live in it.”

The PCFG assigns a probability to each potential parse of a sentence. The highest probability parse is selected; its associated tree is used to derive our measure of sentence complexity. We take the depth of the tree for each sentence in the text and derive three measures per text: the maximum tree depth, the average depth, and the standard deviation of tree depths. As an example, the parse tree in Figure 1 has a depth of 10.

3.3.3.2 Rare Word Use

The number of rare words in a text influences reading comprehension; one rare word can lead to a complete miscomprehension of a sentence [23]. Given this fact, it is important to capture the amount of rare word use in a text when assessing reading comprehension. We model vocabulary rarity by quantifying the percentage of rare words in each text of the test. This language feature is expected to represent the difficulty of the text on a vocabulary level.

To calculate the percentage of rare words, we first used the Children’s Book Test corpus [27] to compute all word frequencies. We then chose a cut-off threshold (700) for determining word rarity. The top 700 frequent words occupy 1.5% of all distinct words and account for 60% of the corpus content. The most frequent 700 words were extracted to form a common word list. We consider words rare if they are not contained in the common word list.

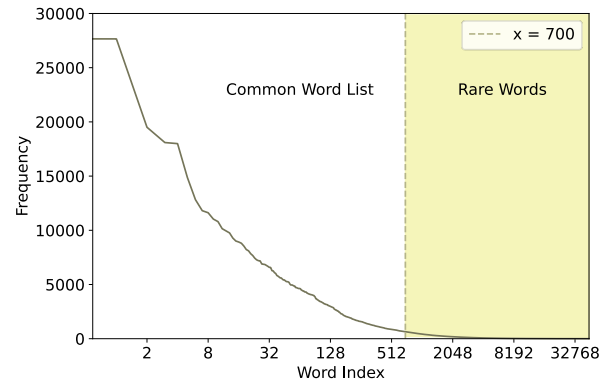


Figure 2. Word frequency in the Children’s Book Test dataset. The x-axis is in \log_2 scale. The dashed line is the cut-off at 700; any words with indices that exceed the cut-off are excluded.

To compute the percentage of rare words in a text, we count the number of rare word tokens (i.e., those that do not appear in the common word list) and divide that by the total number of word tokens in the text. This percentage is our measure of rare word use.

$$\text{Rare Word Use} = \frac{\text{rare word count}}{\text{total word count}}$$

3.3.3.3 Lexical Surprisal

The surprisal of a word in a sentence is related to the amount of cognitive work required for human comprehension of that word within the sentence [33], and it is predictive of reading times [46]. To model this word-level feature, we needed to determine the likelihood of one word appearing right after another word.

Since our prediction task is targeted towards Grade 3 students with elementary-level vocabulary, we built bigrams (n-grams of length 2) from the Children’s Book Test dataset [27]. We then derived the lexical surprisal value [25] for each sentence in the Gates-MacGinitie texts by computing the mean word-level surprisal. Min-max normalization [42] was used to ensure the values are on a similar scale to the other features. After extracting the surprisal of each

sentence in the text, we compute the mean, standard deviation, and maximum of all sentence surprisal values for the whole text. These normalized values are used for the lexical surprisal input feature.

3.3.3.4 Sentence Context

The sentence-context feature employs a prevalent method for representing text - a universal sentence encoder – which incorporates context, meaning, sentence complexity, and word order information [6]. The embedding represents the text in the form of a vector. To obtain these embeddings, we used the pre-trained MPNet encoder [49], which is a transformer-based language model. MPNet was used to encode both the text and questions into vectors with 768 dimensions. The generated universal sentence embedding was used to represent the general context of the texts.

3.4 Prediction Task

The goal of each model is to predict whether a particular student would answer a specific reading comprehension question correctly or incorrectly based on measures of their individual knowledge and sub-skills and the features of the texts (see Figure 3).

The input to our task includes features derived from the textual content of the Comprehension subtest of the Gates–MacGinitie Reading Tests and students’ individual knowledge and sub-skill measures that provide information about their oral language skills. Since each of these instruments provides scores on different scales, we applied min-max scaling normalization [41]. This allows all measures to be placed on the same scale, thereby facilitating comparisons. This type of normalization also facilitates model accuracy in classification tasks [4].

Of the questions students answered, 54 % are labelled correct and 46 % incorrect. Due to the limited size of our data (6,576 entries), all entries were used during model training and testing.

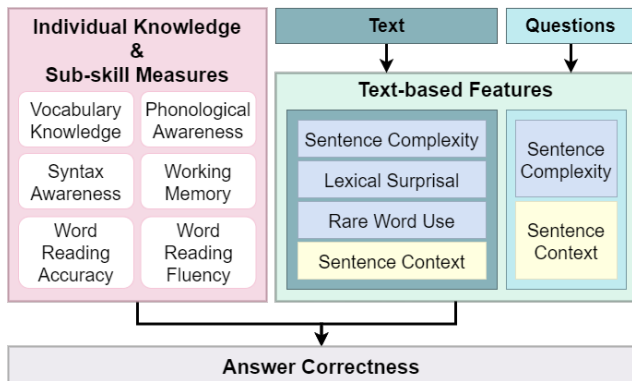


Figure 3. The features used in our prediction task, grouped by measure (child) and feature (text) source.

3.5 Models

3.5.1 Base Model

The Base Model is a fully connected neural network classifier that takes all sub-skill measures as input to predict whether a student will answer each question correctly. Given that the input layer is relatively small with 6 features, our network has two hidden layers each with 12 nodes.

3.5.2 Text-based Feature Models

Our proposed models augment the Base Model with the language features of the text. They incorporate four features: sentence complexity, vocabulary rarity, word surprisal, and context.

3.5.3 Base + Sentence Complexity Model

This model is a fully connected neural network that includes all sub-skill measures and sentence complexity text features as input. The network has two hidden layers, both of which have 18 nodes and use the ReLU activation function. Since we are predicting the probability of a binary event (correct/incorrect), a sigmoid layer follows the hidden layer. Loss is calculated using binary cross entropy. Adam optimization [32] is also used.

3.5.4 Base + Rare Word Use Model

This model is a fully connected neural network that includes all sub-skill measures and rare word use as input features. The network has two hidden layers, both of which have 14 nodes and use the ReLU activation function. Following the hidden layer there is a sigmoid layer. Loss was calculated using binary cross entropy and optimization was done using Adam.

3.5.5 Base + Lexical Surprisal Model

This model is a fully connected neural network that includes all sub-skill measures and lexical surprisal features as input. The network has two hidden layers, both of which have 18 nodes and use the ReLU activation function. Following the hidden layer, there is a sigmoid layer and the loss was calculated using binary cross entropy. Adam was used for optimization.

3.5.6 Base + Context

This model, shown in Figure 4, is a neural network that includes two parts, sub-skill measures (Base Model) and context as represented through sentence embeddings.

The six sub-skill measures are first passed through a 6x12 dense layer.

The sentence context part starts with two vectors: one represents the embeddings from the text and the other that of the question. Each vector is then passed through a 764x64 dense layer and transformed into 64 nodes.

We concatenate the sentence embedding and sub-skill measures into one fully connected layer followed by a hidden layer with 128 nodes. A sigmoid layer follows the hidden layer. Once again, loss was calculated using the binary cross-entropy function, and Adam optimization was performed.

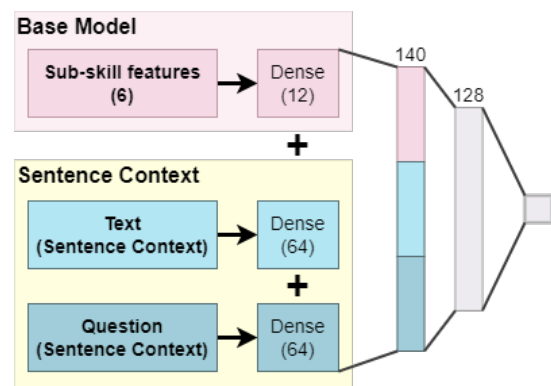


Figure 4. Neural network structure for the Base + Context Model. The “+” indicates the concatenation operation from the previous three dense layers.

3.5.7 Full Model

The Full Model (Figure 5) consists of all sub-skill measures from the Base Model, sentence complexity, vocabulary rarity, word

surprisal, and sentence context. This should allow us to best capture the language features of the tests that might affect children’s reading comprehension performance.

We manually tuned the number of nodes in each layer. Specifically, for the dense layer after concatenating sub-skill and text-based features, we tried {12, 24, 48, 96}; 24 nodes had the best performance. For the dense layers after vector embedding, we tested values among {32, 64, 128, 256} and picked 64. Both dense layers from the vector embeddings are kept at the same size, and no individual testing was done. For the last layer, where all three hidden layers were combined, we tested among {32, 64, 128, 256} for number of nodes and 128 nodes yielded the best result.

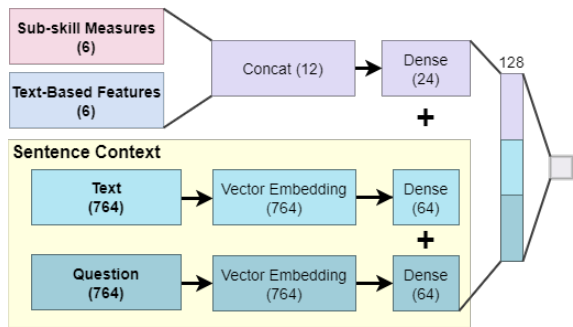


Figure 5. Neural network structure for the Full Model, where all measures and features are used as input. The “+” indicates the concatenation operation from the previous three dense layers.

3.5.8 Hyperparameter Tuning

For all of our models, we tested three learning rates: .01, .001, and .0001. The learning rate of .001 returned the best result for all models. Table 3 shows tuning details for the number of nodes in the two dense layers that are part of our models. We kept the node number the same for both dense layers during tuning.

Table 3. Hyperparameter tuning details

Model	Values Tested	Value Chosen
Base	6, 12, 24	12
Base + Sent. Complex.	9, 18, 36	18
Base + Context	64, 128, 256	128
Base + Rare Word Use	7, 14, 28	14
Base + Lexical Surprisal	9, 18, 36	18

Note. Sent. – Sentence, Complex. – Complexity

3.6 Model Evaluation & Analysis Procedures

We used k-fold cross-validation with $k = 10$ to evaluate all models. Since predicting whether a student answers a question either correctly or incorrectly is equally important, all models were optimized for accuracy during hyperparameter tuning. The whole training process is done within approximately two hours on a commodity machine after hyperparameter tuning.

We used McNemar’s test to determine whether there is a significant difference between the Base Model and all other models. McNemar’s test compares machine learning based classifiers in pairs by looking at the correct and incorrect classification each model makes. It can distinguish model differences even if the models yield similar accuracy results [37]. Bonferroni step-down correction was used to control for multiple comparisons; the adjusted p-values are reported.

Since it is not enough to determine which model performs best, we used integrated gradients [50] to examine feature importance. Integrated gradients quantitatively shows the effect each feature has on the final prediction. This measure evaluates the contribution of features (feature importance) to the prediction results of machine learning models. It does so by gradually increasing the values of the input features from those of a baseline. We used the value zero as our baseline since it represents zero information in our study. This method computes the integral of gradients with respect to inputs along the path from outputs to inputs in a neural network. Integrated gradients can provide insight into which features contributed to students correctly answering a question. A positive integrated gradients value shows an increase in the feature value that contributes to an increase in the output prediction value. A negative value indicates the opposite. The magnitude of the value shows how much the input feature influences the output. A value of zero means the measure or feature did not contribute to the prediction. There is no specific range for integrated gradients because the slope describing the relationship between the input and output (gradient) is not bounded.

4. RESULTS

4.1.1 What is the added value of including text features as input when predicting the correctness of question answering in reading tasks?

Model performance was measured through precision, recall, and accuracy (see Table 4). Chance accuracy for this prediction task was 54%. The average training and validation accuracy difference for all models during cross validation is under 2%.

Table 4. K-fold (k = 10) cross-validation results for each model as $M(SD)$.

Model	Precision	Recall	Accuracy (%)
Base	.665 (.012)	.737 (.011)	65.8 (0.76)
Base + Sent. Complex.	.696 (.017)	.751 (.015)	67.5 (1.17)
Base + Context	.713 (.014)	.738 (.017)	68.7 (0.91)
Base + Rare Word Use	.704 (.009)	.751 (.019)	68.1 (0.69)
Base + Lexical Surprisal	.691 (.014)	.756 (.019)	67.6 (0.74)
Full	.715 (.016)	.747 (.013)	69.8 (0.95)

Note. Sent. – Sentence, Complex. – Complexity

The full Model yields the highest accuracy and precision. It also had relatively high recall, suggesting the Full Model performs well. The Base Model had high recall but the lowest precision and accuracy. Adding any one of sentence complexity, rare word use, or surprisal to the Base Model was associated with an increase in precision while maintaining similar recall, demonstrating the potential for text-based features to support model prediction. Adding sentence embeddings to the Base Model resulted in the highest precision among all models except for the Full Model. However, it also had the lowest recall value. Thus, the model that included context, as represented through embeddings, made the lowest number of errors in its predictions of answering a question correctly. However, it also had the most prediction errors for questions that students had answered incorrectly.

The Full Model and Base + Context Model show significantly higher predictive accuracy than the Base Model (see Table 5). The performance of the remaining models did not show significant differences from that of the Base Model.

Table 5. Model comparisons using McNemar’s Test

Model 1	Model 2	McNemar Test Statistic	<i>p</i>
Full Model	Base Model	7.771	.025
Base + Context	Base Model	8.040	.024
Base + Sentence Complexity	Base Model	0.547	1
Base + Vocabulary Rarity	Base Model	2.972	.340
Base + Lexical Surprisal	Base Model	0.105	.746
Full Model	Base + Context	4.971	.026

Since the full model and Base + Context Model both outperformed the Base Model, we compared these top-performing models to see whether one outperformed the other. While the difference is

relatively small, the Full Model achieves higher performance than the Base + Context Model suggesting that representing many aspects of the text is important.

4.1.2 Full Model Features

We used integrated gradients to examine how much each text-based feature affected whether the model would predict that a student answered a question correctly. In Figure 6, the y-axis shows the value of integrated gradients for each feature, with large values indicating the feature has a strong influence on the model’s prediction. A positive value indicates the feature contributes to predicting a question is answered correctly. Negative values mean the feature supports the prediction of incorrect student answers.

The integrated gradient results for the Full Model show that an increase in maximum sentence complexity and mean question complexity helps the Full Model predict when students will be more likely to incorrectly answer a question (Figure 6). In contrast, predicting when students will correctly answer a question is supported by an increase in maximum (max) sentence complexity. Contributions from all other features are low.

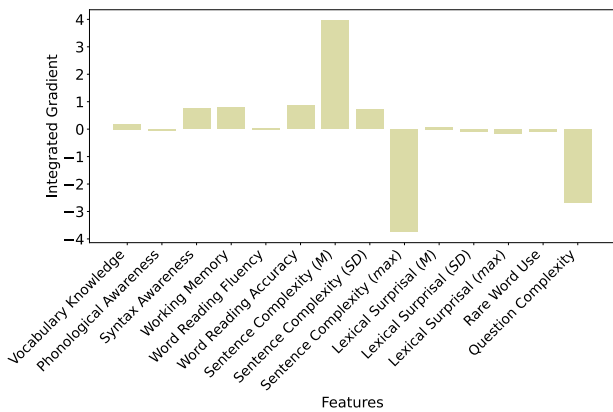


Figure 6. Integrated gradient result for each feature from the Full Model. *M* – mean and *SD* - standard deviation

4.1.3 Sentence Complexity Features

Here we investigate how much the mean, standard deviation, and maximum values of sentence depth contributed to the prediction in the Base + Sentence Complexity Model.

As can be seen in Figure 7, the integrated gradient results do not show substantial contributions from any of the features when they are used on their own. To see if this finding is due to invariability among different instances, we plotted the feature value distribution across all texts (Figure 8). As shown by the values for texts two,

six, and eight, which are relatively high, a lack of variability fails to account for the limited contribution of this feature.

4.1.4 Lexical Surprisal Features

Here we investigate how much the mean, standard deviation, and maximum values of lexical surprisal contributed to the prediction when the Base + Lexical Surprisal Model was used. The integrated gradients results show both the mean and maximum of lexical surprisal contributed little to the prediction (Figure 9). Mean has a small positive impact while maximum has a small negative impact. Given the relatively small gradient, we checked variability across texts: Figure 10 shows that the mean lexical surprisal is relatively stable (its values range from .02 to .04) whereas the maximum lexical surprisal values vary considerably (~.02 - .12).

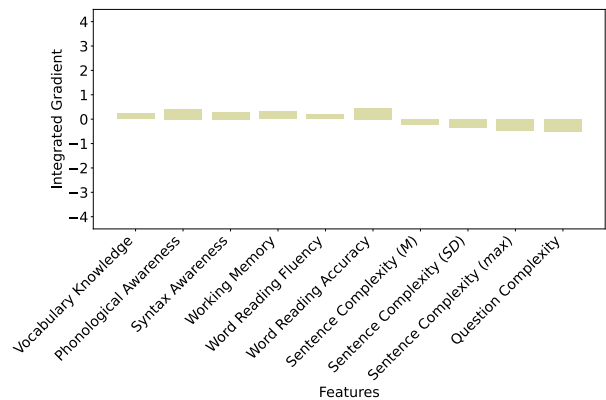


Figure 7. Integrated gradient result for each feature from the Base + Sentence Complexity model. *M* – mean and *SD* - standard deviation

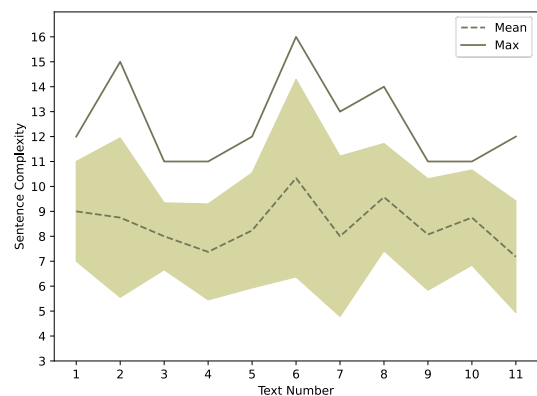


Figure 8. Maximum and mean sentence complexity values for each text. The band represents the area within one standard deviation of the mean.

4.2 Error analysis

We compare the models that use text features as input to the Base Model because the Base Model does not incorporate any text features. Thus, it is not biased towards or against any of the readings or questions. By comparing model performance against that of the Base Model, we can see where a particular feature supports the task and where it performs poorly.

Base + Rare Word Use Model performed especially poorly in Text 7 compared to the Base Model. Text 7 is an expository paragraph containing scientific facts about snow colors. The rare word use feature of Text 7 has a high value of .550 where the average rare word use value across all texts (including Text 7) is .493. The most errors occur for question 30 in Text 7, which is an inference question where the answer is not explicitly stated in the text.

Among the questions where the Base + Rare Word Use Model outperformed the Base Model, Text 8 has the highest percentage of correct predictions. Questions 34 and 35 from Text 8 have a particularly high correct rate and rare word use rate. Both questions are non-inference questions. These differences in performance indicate that this model performs well in situations when the vocabulary is difficult and questions are direct.

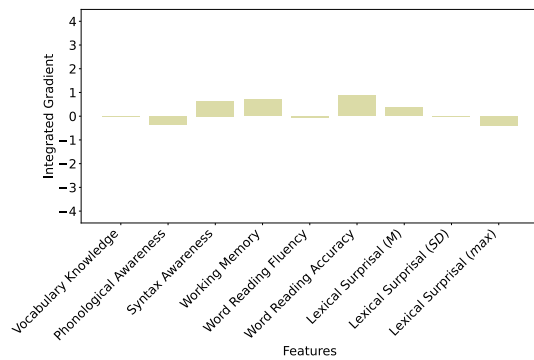


Figure 9. Integrated gradient result for each feature from the base + lexical surprisal model. *M* - mean and *SD* - standard deviation

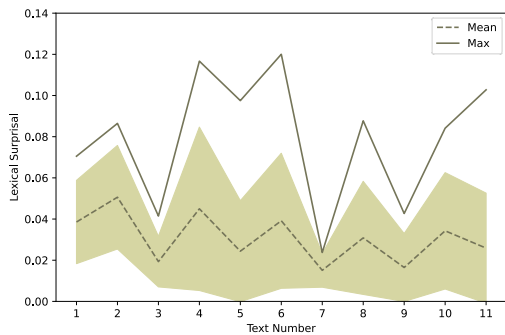


Figure 10. Mean and maximum lexical surprisal values for each text. The band represents the area within one standard deviation from the mean

Base + Sentence Complexity Model output differs in many places from that of the Base Model for Text 7. When comparing these two models on the Text 7 questions, we see that it is common for one of the models to correctly predict student performance when the other is incorrect. All three questions that belong to this text have high error rates in classification. The mean depth feature of this text is 10.3 whereas the average depth across all texts is 8.4.

Base + Context Model predictions disagreed often with those of the Base Model for Text 11, questions 44, 45, and 48. Question 44 and 45 are inference questions and question 48 uses a near synonym for a keyword from the text to assess comprehension. The Base + Context Model performed particularly well in Text 8, question 31 and 34; the answers to these questions can be explicitly found in the text. This differential performance suggests that adding sentence context information supports model performance when predicting student responses to non-inference questions.

Full Model prediction errors come from questions 44 and 45 from Text 11 in most cases. These errors are similar to those made by the Base + Context Model, but the Full Model performs slightly better: the percentage of prediction errors decreased from 22.5% to 18.5% (question 44) and 19.3% to 16.9% (question 45). The Full Model performed particularly well on Text 4, question 14 and 16. Text 4 is a relatively short narrative text containing sentences with simple structures, and the answer to both of the Text 4 questions is easy to find in the text.

5. DISCUSSION

To answer the research question, “What is the added value of including text features as input when predicting the correctness of question answering in reading tasks?”, we trained and evaluated models using information about students’ oral language skills and features from the texts they were reading as input. The developed models were used to explore how these features supported the prediction of student performance on reading-comprehension questions. The text-based features each provide a hidden representation of aspects of the text that were expected to affect reading comprehension. Individually, their inclusion does not appear to lead to strong model performance; however, their joint use supports prediction across a variety of text and question types.

It appears that the strengths identified when adding some features compensate for the weaknesses associated with adding others. For example, sentence complexity performs poorly when the vocabulary is challenging, which can be captured through the addition of the rare word use feature. Similarly, the context feature underperforms on inference questions, which seem to be better supported by the combined use of all features. Analyses of feature contributions to model performance (Section 5.1.2), and the error analysis (Section 5.2) indicate that the features interact. The complicated nature of how features supported the prediction task suggests that text characteristics need to be treated in a nuanced manner if we are to understand children’s reading comprehension or develop learner models that can be used to adapt reading materials to children.

5.1 Text-based Feature Contributions to Prediction

Each of the text features that we added appeared to improve model accuracy when they were used jointly (Full Model). The only text feature that supported improved model performance on its own was the sentence context feature that used sentence embeddings, and this model was outperformed by the Full Model that included sentence context as one of its text-based features. Given the relatively high contribution of sentence complexity features to the Full Model and the low contribution of most other text-based features (as identified using integrated gradients - Section 5.1.2), the sentence context and other features likely augmented the information provided by the sentence complexity feature.

The improved performance associated with adding sentence context information may be the result of it implicitly capturing several aspects of the text. Universal sentence encoders, which were used

to extract the sentence context feature, incorporate word order, sentence meaning, and grammar into a single vector so they implicitly capture some aspects of sentence complexity and vocabulary. They also implicitly capture aspects of the content domain through their representation of the surrounding text. The support provided by adding the sentence context feature parallels Hirsch's [28] findings on reading comprehension of elementary students. Hirsch found that a gap in reading comprehension appears (or is widened) when more advanced domains (e.g., math, science) are introduced. This effect of domain context was also found to be more influential when compared to differences in vocabulary size and decoding skills among students in elementary school. Our models accounted for the vocabulary knowledge (i.e., PPVT-M score) of children and added text-based features that represent the vocabulary knowledge needed to understand the texts (i.e., rare word use). This text-based feature only contributed to improved model performance when used in combination with sentence context. This lack of contribution by rare word use on its own suggests that Hirsch's domain influence findings can be partly captured through the sentence context feature.

The sentence complexity of both the text and the question contributed substantially to predictions when sentence complexity was used alongside other features even though this feature did not improve model performance when added to the Base Model by itself. This pattern suggests the sentence complexity of both the text and the question interacts with other features to produce meaningful predictions. That is, sentence structure by itself does not determine the difficulty students may experience when trying to understand a piece of text. This is consistent with our error analysis (Section 5.2) which identified high error rates for questions with low levels of rare word use and high levels of mean sentence complexity.

The rare word use feature improved prediction task performance (section 5.1.1), which is consistent with the lexical quality hypothesis [44]. Lexical quality is typically interpreted in relation to the role of high-quality word representations in reading comprehension; the flip side is that the presence of rare words will negatively impact reading comprehension because of their connection to the precision and flexibility of a mental representation of a word. Our rare word use feature provides a way to model the demands that a text will place on lexical quality, which will in turn be related to reading comprehension. That said, it seems that the most frequent prediction error emerged for inference questions accompanying a scientific information text with high levels of rare word use.

One's knowledge of vocabulary and exposure to language influence what one expects to see. Like with rare words, people can be surprised by the use of a relatively familiar word in an unexpected collocation or context [23]. The lexical surprisal feature was meant to capture this element of how we process language. In contrast to our expectations, the n-gram language model that we used to measure lexical surprisal did not appear to help predict performance on reading comprehension questions, either on its own or in combination with other text-based features. Its lack of contribution to predicting student correctness may be partly due to the limited diversity of content in the training corpus. For future work, the Corpus of Contemporary American English [9] is another publicly available dataset that could be used to model surprisal for children's readings. Additionally, using other language modelling approaches, such as a PCFG [46] or recurrent neural network grammars [26], to obtain surprisal measures might yield more accurate predictions since these language modelling approaches can output a more nuanced estimate of surprisal. It is also possible that rare word use and sentence context capture this aspect of language

processing for the target text. This suggests a need for follow up work that can help to tease apart the contributions of these vocabulary-related features.

Like the Base + Context Model, the Full Model is also less accurate when predicting correctness for inference questions. This shortcoming is mitigated by the inclusion of other text-based features that supported the prediction task. The analysis of feature contributions to this model (see Section 5.1.2) shows that the features that supported the prediction of a correct student answer (the mean and standard deviation of sentence complexity in a text) were different from those used to inform the prediction of which questions would be answered incorrectly (the maximum sentence complexity in a text and the sentence complexity of the question). This finding parallels those of Perin and Lauterbach [45], who found that there was consistency in the features that predicted strong performance in student writing and that different features predicted low student scores. The fact that none of the oral language skills captured through the individual knowledge and sub-skill measures supported the prediction of which questions a child would answer incorrectly suggests that Perin and Lauterbach's argument for there being many ways to perform poorly and only a few to perform well might be extendable to reading comprehension. When combined with the ability of text-based features to predict lower performance, this finding suggests text-based measures are needed to fully capture information about the reading-comprehension challenges that children face.

5.2 Implications for Understanding Reading

In addition to needing text-based features to predict when students will incorrectly answer a question, the error analyses showed that different text-based features were useful for predicting answer correctness for each type of reading comprehension task (inference, retrieval from text). This finding implies that we need to consider different features of a text when trying to understand children's inferring abilities or their ability to identify the content that is relevant to a question.

Beyond demonstrating the potential utility of these text-based features for predicting student answers to reading comprehension questions, these models and the approaches used can provide insight into the text-related challenges that affect students' reading comprehension based on their knowledge levels. Among these are the vocabulary used and whether its use matches that expected by a child based on their exposure to the language. It was expected that surprisal would support this prediction task since it has been predictive of reading times in some studies [46]. However, surprisal contributed little to model prediction, suggesting a need to further study the role of surprisal in children's processing.

The different contributions of features to the prediction task (see Figure 6), suggest that the sentence context feature augments information provided by the sentence complexity feature. Students tended to have poor comprehension when there was even just one very complex sentence, and higher mean sentence complexity showed less effect on the prediction result. This indicates that children's ability to answer a reading comprehension question is less affected when the sentences are more complex overall, compared to having a particularly complex sentence. The specific contributions of the mean, standard deviation, and maximum values of this characteristic of the text imply that all three measures should be used to better understand student reading abilities in relation to a text. Including the maximum sentence complexity helps to understand when students answer a question incorrectly and the mean and standard deviation help to understand when a question is

answered correctly. Combined, these provide a more robust view of how learners comprehend text.

5.3 Implications for Learner Modelling and Adaptation

As reported in our results, adding the sentence context feature to the Base Model supported better predictions of correct responses to a comprehension question. The sentence context feature also produced a better performing model when combined with the other text-based features. The underperformance of the Base + Context Model relative to the Full Model may be due to embeddings not having explicit representations of sentence complexity and lexical surprisal. The Base + Context Model's lack of explicit representation of these specific linguistic traits may be why our Full Model had the best performance.

The performance of the Full Model suggests that we can augment the learner modelling and adaptation process in educational technologies by using features that are automatically extractable from texts. This implication is further supported by the performance of the Base + Context model. The sentence context feature relies on a heavily data-driven approach to support prediction for non-inference questions that share a similar context with the text. Since the addition did not support the ability to distinguish student performance on inference questions, additional mechanisms will need to be found to support prediction for inference-based questions, which are usually harder for students to answer than non-inference questions [48]. In our case, the inclusion of multiple text-based features, specifically rare word use, helped mitigate the limitations of the context feature when inference-based questions were being predicted. So, this strategy can be used until a more powerful feature is found.

As suggested by the insights gleaned from the sentence complexity measures, these features should be included to better identify which questions might be within a student's abilities and which might not. This suggestion builds on Scott's findings from a study investigating sentence comprehension [47]. In both Scott's and our setting, measures of sentence complexity were more effective in prediction within the context of a specific domain. This may also suggest the benefit of including our sentence context features as they could reinforce domain information while also providing some information about sentence complexity and vocabulary use.

It is worth noting that, as part of the nature of all universal encoder models, the generated embedding is less interpretable by people even though it provides an effective representation of the text for predicting comprehension performance. This means more effort is required to evaluate how much specific features of the text affect the prediction task. Given the enhanced performance that is associated with the use of these embeddings, this effort is warranted when developing models for supporting the adaptive selection of texts and questions in an educational technology.

6. CONCLUSION

Student performance on reading comprehension tasks is often predicted using assessments of oral language skills, such as vocabulary knowledge or syntactic awareness. We extend this work by capturing features of text, which are rarely used in the prediction process despite an understanding that the characteristics of a text influence the ability to understand that text. In the present paper, we report

on data from grade 3 students to develop and test our prediction model. We test automated methods for extracting and incorporating text-based features from the content of a reading test to help predict whether a student will answer a reading comprehension question correctly.

The extracted text-based features were selected based on theories or evidence supporting their relationship with reading comprehension. Thus, they were designed to provide information about some of the aspects of a text that are expected to interact with children's oral language skills. Specifically, the selected features represent sentence complexity, vocabulary frequency, lexical surprisal, and context. They were all extracted using natural language processing techniques that include n-grams, a probabilistic context-free grammar, and a universal sentence encoder.

We used these features as inputs to neural networks that can self-update when given more student data or new texts are added. This mitigates the disadvantage where models with no text-based features have to be retrained fully every time a test is updated. Our model is also expected to better handle diversity across reading comprehension tests and student reading skill levels because it relies on general features of a text rather than solely relying on historical records of student performance. Since our model has clear separation between different categories of features at the input layer, new features and new data can be easily added which offers a starting point to identify what additional training might help improve student reading skills. To support the continued study of the role that text features play in reading comprehension, we have shared our approach to interpreting and analyzing the model (in section 5.4 and through GitHub²). This sharing will allow others to apply this approach to another test or population of learners.

To summarize, we extracted and tested text-based features at the word and sentence level to examine their impact on the reading comprehension of children. Those features were used as input to a model that predicts student performance on reading comprehension tests at the question level. Analyses of the tested models show that one of the employed text-based features (e.g., sentence context) improved model performance on its own while others did not (e.g., sentence complexity and lexical surprisal). The joint use of the text-based features resulted in a more than 4 % gain. Subsequent error analysis suggests each of these text-based features represents an important characteristic of the text, with their combined use resulting in the best performing model. Exploration of how these text-based features contributed to model performance provided insight into the complex relationships between text features and children's reading comprehension performance. Thus, the models and their analysis can support the design of better learning systems, the selection of appropriate reading materials, and increased understanding of the multi-faceted nature of student reading comprehension.

7. ACKNOWLEDGMENTS

This work is supported in part by funding from the Social Sciences and Humanities Research Council of Canada and the Natural Sciences and Engineering Research Council of Canada (NSERC), [RGPIN/293300-2013], Canadian Institute for Advanced Research (CIFAR), and Alberta Machine Intelligence Institute (Amii).

² <https://github.com/EdTeKLA/ComprehensionScorePredictor>

8. REFERENCES

- [1] Allen, L.K., Snow, E.L. and McNamara, D.S. 2015. Are you reading my mind?: modeling students' reading comprehension skills with natural language processing techniques. *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge* (Poughkeepsie New York, Mar. 2015), 246–254.
- [2] Beck, J. and Sison, J. 2006. Using Knowledge Tracing in a Noisy Environment to Measure Student Reading Proficiencies. *I. J. Artificial Intelligence in Education*. 16, (Jan. 2006), 129–143.
- [3] Beck, J.E., Jia, P. and Mostow, J. 2003. Assessing Student Proficiency in a Reading Tutor That Listens. *User Modeling 2003*. P. Brusilovsky, A. Corbett, and F. de Rosis, eds. Springer Berlin Heidelberg. 323–327.
- [4] Borkin, D., Némethová, A., Michalčonok, G. and Maiorov, K. 2019. Impact of Data Normalization on Classification Model Accuracy. *Research Papers Faculty of Materials Science and Technology Slovak University of Technology*. 27, 45 (Sep. 2019), 79–84. DOI:<https://doi.org/10.2478/rput-2019-0029>.
- [5] Cain, K. 2007. Syntactic awareness and reading ability: Is there any evidence for a special relationship? *Applied Psycholinguistics*. 28, 4 (Oct. 2007), 679–694. DOI:<https://doi.org/10.1017/S0142716407070361>.
- [6] Cer, D., Yang, Y., Kong, S., Hua, N., Limtiaco, N., John, R.S., Constant, N., Guajardo-Cespedes, M., Yuan, S., Tar, C., Sung, Y.-H., Strophe, B. and Kurzweil, R. 2018. Universal Sentence Encoder. *arXiv:1803.11175 [cs]*. (Apr. 2018). <http://arxiv.org/abs/1803.11175>
- [7] Colenutt, A., Toye, M.A. and College, F. 2012. *Critical Crossroads: Youth, Criminal Justice and Literacy: Literature Review: Transformative Literacy for Youth in Conflict with the Law*. Frontier College.
- [8] Corlatescu, D.-G., Dascalu, M. and McNamara, D.S. 2021. Automated Model of Comprehension V2.0. *Artificial Intelligence in Education*. I. Roll, D. McNamara, S. Sosnovsky, R. Luckin, and V. Dimitrova, eds. Springer International Publishing. 119–123.
- [9] Davies, M. 2008. *The Corpus of Contemporary American English*.
- [10] Deacon, S.H., Benere, J. and Castles, A. 2012. Chicken or egg? Untangling the relationship between orthographic processing skill and reading accuracy. *Cognition*. 122, 1 (Jan. 2012), 110–117. DOI:<https://doi.org/10.1016/j.cognition.2011.09.003>.
- [11] Deacon, S.H. and Kieffer, M. 2018. Understanding how syntactic awareness contributes to reading comprehension: Evidence from mediation and longitudinal models. *Journal of Educational Psychology*. 110, 1 (Jan. 2018), 72–86. DOI:<https://doi.org/10.1037/edu0000198>.
- [12] Demmans Epp, C. 2019. Developing an Adaptive Mobile Tool to Scaffold the Communication and Vocabulary Acquisition of Language Learners. *Handbook of Mobile Teaching and Learning*. Y. Zhang and D. Cristol, eds. Springer Berlin Heidelberg. 1–26.
- [13] Demmans Epp, C. and McCalla, G. 2011. ProTutor: Historic Open Learner Models for Pronunciation Tutoring. *Artificial Intelligence in Education*. G. Biswas, S. Bull, J. Kay, and A. Mitrovic, eds. Springer Berlin Heidelberg. 441–443.
- [14] Demmans Epp, C. and Phirangee, K. 2019. Exploring mobile tool integration: Design activities carefully or students may not learn. *Contemporary Educational Psychology*. 59, (Oct. 2019), 101791. DOI:<https://doi.org/10.1016/j.cedpsych.2019.101791>.
- [15] Devlin, J., Chang, M.-W., Lee, K. and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)* (2019), 4171–4186.
- [16] Duong, M., Mostow, J. and Sitaram, S. 2011. Two methods for assessing oral reading prosody. *ACM Transactions on Speech and Language Processing*. 7, 4 (Aug. 2011), 1–22. DOI:<https://doi.org/10.1145/1998384.1998388>.
- [17] Fang, Y., Li, T., Roscoe, R.D. and McNamara, D.S. 2021. Predicting Literacy Skills via Stealth Assessment in a Simple Vocabulary Game. *Adaptive Instructional Systems. Adaptation Strategies and Methods*. R.A. Sottolare and J. Schwarz, eds. Springer International Publishing. 32–44.
- [18] Frank, S. and Hoeks, J.C.J. 2019. The interaction between structure and meaning in sentence comprehension: Recurrent neural networks and reading times. *CogSci* (2019).
- [19] Gough, P.B. and Tunmer, W.E. 1986. Decoding, Reading, and Reading Disability. *Remedial and Special Education*. 7, 1 (Jan. 1986), 6–10. DOI:<https://doi.org/10.1177/074193258600700104>.
- [20] Graesser, A.C. 2006. Question Understanding Aid (QUAID): A Web Facility that Tests Question Comprehensibility. *Public Opinion Quarterly*. 70, 1 (Mar. 2006), 3–22. DOI:<https://doi.org/10.1093/poq/nfj012>.
- [21] Graesser, A.C., Greenberg, D., Frijters, J.C. and Talwar, A. 2021. Using AutoTutor to Track Performance and Engagement in a Reading Comprehension Intervention for Adult Literacy Students. *Revista signos*. 54, 107 (Dec. 2021), 1089–1114. DOI:<https://doi.org/10.4067/S0718-09342021000301089>.
- [22] Graesser, A.C., McNamara, D.S. and Kulikowich, J.M. 2011. Coh-Metrix: Providing Multilevel Analyses of Text Characteristics. *Educational Researcher*. 40, 5 (Jun. 2011), 223–234. DOI:<https://doi.org/10.3102/0013189X11413260>.
- [23] Graesser, A.C., McNamara, D.S. and Louwerse, M.M. 2003. What do readers need to learn in order to process coherence relations in narrative and expository text. *Rethinking reading comprehension*. Guilford Press. 82–98.
- [24] Guo, W., Cho, B.-Y. and Wang, J. 2020. StrategicReading: Understanding Complex Mobile Reading Strategies via Implicit Behavior Sensing. *Proceedings of the 2020 International Conference on Multimodal Interaction* (Virtual Event Netherlands, Oct. 2020), 491–500.
- [25] Hale, J. 2001. A probabilistic early parser as a psycholinguistic model. *Second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies 2001 - NAACL '01* (Pittsburgh, Pennsylvania, 2001), 1–8.

- [26] Hale, J., Dyer, C., Kuncoro, A. and Brennan, J. 2018. Finding syntax in human encephalography with beam search. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (Melbourne, Australia, 2018), 2727–2736.
- [27] Hill, F., Bordes, A., Chopra, S. and Weston, J. 2016. The Goldilocks Principle: Reading Children’s Books with Explicit Memory Representations. *ICLR* (2016).
- [28] Hirsch, E.D. 2003. Reading comprehension requires knowledge of words and the world. *American educator*. 27, 1 (2003), 10–13.
- [29] Hofmann, M.J., Remus, S., Biemann, C., Radach, R. and Kuchinke, L. 2022. Language Models Explain Word Reading Times Better Than Empirical Predictability. *Frontiers in Artificial Intelligence*. 4, (Feb. 2022). DOI:<https://doi.org/10.3389/frai.2021.730570>.
- [30] Kamil, M.L. 2003. *Adolescents and literacy: Reading for the 21st century*. Alliance for Excellent Education.
- [31] Karita, S., Chen, N., Hayashi, T., Hori, T., Inaguma, H., Jiang, Z., Someki, M., Soplín, N.E.Y., Yamamoto, R., Wang, X., Watanabe, S., Yoshimura, T. and Zhang, W. 2019. A Comparative Study on Transformer vs RNN in Speech Applications. *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)* (SG, Singapore, Dec. 2019), 449–456.
- [32] Kingma, D.P. and Ba, J. 2015. Adam: A Method for Stochastic Optimization. *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings* (2015).
- [33] Levy, R. 2008. Expectation-based syntactic comprehension. *Cognition*. 106, 3 (Mar. 2008), 1126–1177. DOI:<https://doi.org/10.1016/j.cognition.2007.05.006>.
- [34] MacGinitie, W.H., MacGinitie, R.K., Maria, K. and Dreyer, L.G. 2000. *Gates-MacGinitie Reading Tests (4th ed.)*. Riverside Publishing.
- [35] Mao, Y., Lin, C. and Chi, M. 2018. Deep Learning vs. Bayesian Knowledge Tracing: Student Models for Interventions. (Oct. 2018). DOI:<https://doi.org/10.5281/ZENODO.3554691>.
- [36] McKinlay, A. 2011. Peabody Picture Vocabulary Test –Third Edition (PPVT-III). *Encyclopedia of Child Behavior and Development*. S. Goldstein and J.A. Naglieri, eds. Springer US. 1072–1072.
- [37] McNemar, Q. 1947. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*. 12, 2 (Jun. 1947), 153–157. DOI:<https://doi.org/10.1007/BF02295996>.
- [38] Mishev, K., Gjorgjevikj, A., Vodenska, I., Chitkushev, L.T. and Trajanov, D. 2020. Evaluation of Sentiment Analysis in Finance: From Lexicons to Transformers. *IEEE Access*. 8, (2020), 131662–131682. DOI:<https://doi.org/10.1109/ACCESS.2020.3009626>.
- [39] Mitchell, J., Lapata, M., Demberg, V. and Keller, F. 2010. Syntactic and Semantic Factors in Processing Difficulty: An Integrated Measure. *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics. 196–206.
- [40] Nagamani, M., Narendra Prasad, M. and Girija, P.N. 2005. Improving reading and writing skills with INtelligent Tutor for TELugu Language Learning INTTELL. *Proceedings of 2005 International Conference on Intelligent Sensing and Information Processing, 2005*. (Chennai, India, 2005), 426–429b.
- [41] Panda, S.K., Nag, S. and Jana, P.K. 2014. A smoothing based task scheduling algorithm for heterogeneous multi-cloud environment. *2014 International Conference on Parallel, Distributed and Grid Computing* (Solán, India, Dec. 2014), 62–67.
- [41] Patro, S.G.K. and Sahu, K.K. 2015. Normalization: A Pre-processing Stage. *arXiv:1503.06462 [cs]*. (Mar. 2015). <http://arxiv.org/abs/1503.06462>
- [43] Pearson, P.D., Hiebert, E.H. and Kamil, M.L. 2007. Vocabulary assessment: What we know and what we need to learn. *Reading Research Quarterly*. 42, 2 (Apr. 2007), 282–296. DOI:<https://doi.org/10.1598/RRQ.42.2.4>.
- [44] Perfetti, C. 2007. Reading Ability: Lexical Quality to Comprehension. *Scientific Studies of Reading*. 11, 4 (Sep. 2007), 357–383. DOI:<https://doi.org/10.1080/10888430701530730>.
- [45] Perin, D. and Lauterbach, M. 2018. Assessing Text-Based Writing of Low-Skilled College Students. *International Journal of Artificial Intelligence in Education*. 28, 1 (Mar. 2018), 56–78. DOI:<https://doi.org/10.1007/s40593-016-0122-z>.
- [46] Roark, B., Bachrach, A., Cardenas, C. and Pallier, C. 2009. Deriving lexical and syntactic expectation-based measures for psycholinguistic modeling via incremental top-down parsing. *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing Volume 1 - EMNLP '09* (Singapore, 2009), 324.
- [47] Scott, C.M. 2009. A Case for the Sentence in Reading Comprehension. *Language, Speech, and Hearing Services in Schools*. 40, 2 (Apr. 2009), 184–191. DOI:[https://doi.org/10.1044/0161-1461\(2008/08-0042\)](https://doi.org/10.1044/0161-1461(2008/08-0042)).
- [48] Silva, M. and Cain, K. 2015. The relations between lower and higher level comprehension skills and their role in prediction of early reading comprehension. *Journal of Educational Psychology*. 107, 2 (2015), 321–331. DOI:<https://doi.org/10.1037/a0037769>.
- [49] Song, K., Tan, X., Qin, T., Lu, J. and Liu, T.-Y. 2020. MPNet: Masked and Permuted Pre-training for Language Understanding. *Advances in Neural Information Processing Systems* (2020), 16857–16867.
- [50] Sundararajan, M., Taly, A. and Yan, Q. 2017. Axiomatic Attribution for Deep Networks. *Proceedings of the 34th International Conference on Machine Learning* (Aug. 2017), 3319–3328.
- [51] Torgesen, J.K., Rashotte, C.A. and Wagner, R.K. 1999. *TOWRE: Test of word reading efficiency*. Pro-ed Austin, TX.
- [52] Toutanova, K., Klein, D., Manning, C.D. and Singer, Y. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - NAACL '03* (Edmonton, Canada, 2003), 173–180.
- [53] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł. and Polosukhin, I. 2017.

Attention is All you Need. *Advances in Neural Information Processing Systems* (2017).

- [54] Verhoeven, L. and van Leeuwe, J. 2008. Prediction of the development of reading comprehension: a longitudinal study. *Applied Cognitive Psychology*. 22, 3 (Apr. 2008), 407–423. DOI:<https://doi.org/10.1002/acp.1414>.
- [55] Wagner, R.K., Torgesen, J.K., Rashotte, C.A. and Pearson, N.A. 1999. *Comprehensive test of phonological processing: CTOPP*. Pro-ed Austin, TX.
- [56] Wechsler, D. 2003. *Wechsler Intelligence Scale for Children (4th ed.)*. Pearson.
- [57] Wijekumar, K., Graham, S., Harris, K.R., Lei, P.-W., Barkel, A., Aitken, A., Ray, A. and Houston, J. 2019. The roles of writing knowledge, motivation, strategic behaviors, and skills in predicting elementary students' persuasive writing from source material. *Reading and Writing*. 32, 6 (Jun. 2019), 1431–1457. DOI:<https://doi.org/10.1007/s11145-018-9836-7>.
- [58] Woodcock Reading Mastery Tests (Rev. ed.): 1987. .

Enhancing Stealth Assessment in Game-Based Learning Environments with Generative Zero-Shot Learning

Nathan Henderson¹, Halim Acosta¹, Wookhee Min¹, Bradford Mott¹, Trudi Lord²,
Frieda Reichsman², Chad Dorsey², Eric Wiebe¹, James Lester¹

¹North Carolina State University
Raleigh, NC, 27695

{nlhender, hacosta, wmin, bwmott, wiebe, lester}@ncsu.edu

²Concord Consortium
Concord, MA, 01742

{tlord, freichsman, cdorsey}@concord.org

ABSTRACT

Stealth assessment in game-based learning environments has demonstrated significant promise for predicting student competencies and learning outcomes through unobtrusive data capture of student gameplay interactions. However, as machine learning techniques for student competency modeling have increased in complexity, the need for substantial data to induce such models has likewise increased. This raises scalability concerns, as capturing game interaction data is often logistically challenging yet necessary for supervised learning of student competency models. The generalizability of such models also poses significant challenges, and the performance of these models when applied to new domains or gameplay scenarios often suffers. To address these issues, we introduce a zero-shot learning approach that utilizes conditional generative modeling to generalize stealth assessment models for new domains in which prior data and competency labels may not exist. We evaluate our approach using observed student interactions with a game-based learning environment for introductory genetics. We use a conditional generative model to map latent embeddings of genetics concepts and student competencies to student gameplay patterns, enabling the generation of synthetic gameplay data associated with concepts and game levels that have not been previously introduced. Results indicate the zero-shot learning approach enhances the performance of the competency models on unseen game levels and concepts, pointing to more generalizable stealth assessment models and improved prediction of student competencies.

Keywords

Stealth Assessment, Game-Based Learning, Zero-Shot Learning, Student Modeling

N. Henderson, H. Acosta, W. Min, B. Mott, T. Lord, F. Reichsman, C. Dorsey, E. Wiebe, and J. Lester. Enhancing stealth assessment in game-based learning environments with generative zero-shot learning. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 171–182, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852942>

1. INTRODUCTION

Game-based learning has been shown to be effective at promoting student engagement and fostering enhanced learning experiences [7, 31]. These environments can complement traditional learning methods and help students acquire “21st century skills” such as digital literacy, creative thinking, and knowledge acquisition [4]. Further, game-based learning environments can enable educators to unobtrusively analyze student behavior through stealth assessment for the purpose of improving learning outcomes [9, 35]. Stealth assessment, offers a systematic approach for constructing data-driven models of student performance derived from evidentiary arguments [6, 29]. Despite these promising benefits, data-driven student modeling techniques are growing in complexity and often require large amounts of training data, which poses significant challenges.

Collecting sufficient data for training student models is often time and resource intensive, raising scalability concerns for stealth assessment frameworks [45]. Practitioners may also find that modeling student behavior in new domains, educational contexts, and populations is infeasible due to data sparsity issues. Further, circumstances may arise where there is no prior data to train stealth assessment models. Examples include where post-test surveys are impractical to administer, such as informal learning environments like museum exhibits, or if a learning concept or in-game problem-solving task is being deployed for the first time. These problems pose significant practical challenges for stealth assessment models.

Few-shot learning has been introduced as an effective method for classification tasks where labeled data may be scarce for certain classes or tasks [11, 23, 26, 45]. In particular, zero-shot learning (ZSL) refers to scenarios where no samples of a specific class are present at training. ZSL forms a mapping between the data and class labels present at training (“seen” data) and data absent from the training set (“unseen” data) using other attributes (semantic data) to bridge the gap between these two domains. ZSL can also address the aforementioned issues with stealth assessment by generating competent augmented data representative of the “unseen” classes, maintaining intra-class variance while promoting inter-class discrimination based on semantic relationships within the data [11, 45]. This allows for effective data augmentation that can be used to train downstream classifiers to make accurate inferences on data samples from new or unseen classes. Because these techniques

are designed to work in the absence of training data for new or unseen classes, they can be used to bootstrap new models and help mitigate the “cold start” problem where models make poor inferences due to this lack of class-specific training data [46].

In this work, we propose a generative zero-shot learning approach to improve stealth assessment models for predicting student competencies for certain gameplay levels and educational concepts missing prior student interaction data. We utilize conditional generative adversarial networks and embedding representations of introductory genetics concepts to learn latent mappings between student competencies and gameplay behaviors. By generating synthetic gameplay data conditioned on genetics concept descriptors, the generalizability of the competency models can be enhanced, leading to improved predictive performance. Our approach is evaluated using *Geniventure*, a game-based learning environment for teaching introductory genetics concepts to middle and high school students. Stealth assessment models induced with augmented student gameplay data are used to assess our generative ZSL approach. There are few zero-shot learning methods applied within educational domains [11, 22, 45]. However, to our knowledge there has not been prior work on addressing the unseen class problem in stealth assessment. We demonstrate that our approach is an effective method for addressing challenges with data sparsity and unseen class labels for student competency models in a game-based learning environment. Finally, we further show that our approach leads to improvement in the predictive performance of student competency models when compared against non-augmented baselines and alternative generative modeling techniques.

2. RELATED WORK

This work lies at the intersection of zero-shot learning and stealth assessment, particularly addressing approaches to student competency prediction for a range of genetics concepts, which is a subset of student modeling. We provide an overview of recent work pertaining to student modeling in game-based learning, with a focus on stealth assessment. Additionally, we provide a review of recent zero-shot learning research pertaining to student modeling, in addition to zero-shot learning using generative modeling approaches.

2.1 Student Modeling in Game-Based Learning

Student modeling has been shown to be effective at predicting complex learning processes such as engagement, flow, and the incubation effect [21, 32, 40]. More specifically, student modeling within game-based learning has been shown to positively impact deep learning and higher order thinking [19], promote self-regulated learning during gameplay [20, 34, 42], and address cognitive, affective, and social factors for fostering enhanced learning outcomes [3, 16]. Additionally, student modeling within game-based learning environments has been used to analyze and predict student competency levels in a variety of domains, including physics [41], literacy [40], and computational thinking. Student modeling in educational games has also been used to enable personalized and adaptive learning environments [5]. Spaulding et al. investigate the efficacy of transfer of learned cognitive models between two game-based learning environments [40]. The authors investigate the issue of negative transfer by utilizing Gaussian processes and an instance-weighting approach that considers the similarity between source and target tasks. Additionally, they address the aforementioned “cold-start” problem with a multi-task learning approach. Other studies have explored data-driven approaches for inferring student competencies by modeling their in-game progression through activity log data [12]. Similar data-driven approaches have

attempted to design student learning profilers that can inform practitioners in their design of adaptive gamified learning environments tailored to students’ interests and needs. One example is SPOnTo, a student profile ontology, that employs a multi-phase pipeline for student classification [29]. By predicting self-reported student type, intelligence type, and learning difficulties, the authors’ approach shows promise for personalized learning systems enabled by student behavioral patterns.

2.2 Stealth Assessment

There has been increased emphasis on developing effective and robust stealth assessment frameworks in recent years. By varying multiple input features (e.g., ECD model, sample size, data normality significance levels), Georgiadis et al. find that Gaussian Naïve Bayes and C4.5 models were effective for use in stealth assessment and were capable of handling different data distributions with extreme non-normality [14]. However, these models’ accuracy degrades as ECD models increase in complexity. Shute and Rahimi utilize stealth assessment to establish a link between creativity and the properties of well-established learning games that foster creative behavior and propose a creativity criterion [36]. Using Physics Playground and Bayesian networks, they show that their stealth assessment framework offers a valid measure for inferring creativity and was significantly correlated with other performance-based measures of creativity. In contrast to traditional approaches, deep learning-based methods such as long-short term memory networks (LSTMs) have been utilized as a method to model long-term temporal dependencies within student gameplay behaviors [18]. Min et al. employ LSTMs and n -gram based feed forward neural networks and compare their performance to competitive baselines [24]. By combining students’ pre-learning measures and interaction log data from a game-based learning environment, the LSTM-based approach outperformed both baseline methods and the highest performing FFNN using early prediction metrics. Akram et al. achieve similar results supporting the effectiveness of LSTMs as a student modeling technique [1].

2.3 Zero-Shot Learning

Zero-shot learning, first introduced as “zero-data learning” [23], considers the task of recognizing new classes whose instances may not have been seen during training. Recent advances in ZSL have largely been applied in the image and video classification domains, but relatively little work has explored its effectiveness in learning analytics. Wu et al. introduce the “ZSL feedback challenge” utilizing a dataset of 8 assignments with 800 unique solutions to propose a method for attributing feedback to specific sections of student code and to trace knowledge over time [45]. The authors achieve optimal performance by combining a rubric sampling technique with a multimodal variational autoencoder. Their framework can effectively track student growth over time and can provide feedback on non-compiled programs. In an alternative approach, Efremov et al. apply neural program synthesis, a reinforcement learning approach, to generate feedback and step-by-step hints for students from a partial solution [11]. They incorporate abstract syntax tree representations of student code with a tree-based bi-directional LSTM architecture to encode students’ inputs. The learned policy network outperforms state-of-the-art methods such as a continuous hint factory (CHF) and can provide feedback on specific lines of code.

Generative models are another method for approaching the ZSL problem. Mishra et al. introduce a conditional variational autoencoder conditioned on a class embedding vector to reduce domain shift across unseen classes [26]. The generative model is used to



Figure 1. Example challenges in *Geniventure* for the six gameplay levels.

produce synthetic training data that is utilized by a downstream classifier. Their generalized zero-shot learning approach is applied to five popular image recognition datasets and is able to achieve state-of-the-art performance using top- k and per class accuracy.

We contribute to this line of research by introducing a zero-shot learning approach to improve the generalizability of stealth assessment models. This work employs conditional generative adversarial networks for creating synthetic gameplay data from concepts for which there was no previous student interaction data available at training. We demonstrate that this method can improve predictive performance of student mastery within unseen game levels and educational concepts, highlighting its potential as a method for generalizing student competency models.

3. STUDENT GAMEPLAY DATA

Our generative ZSL method is implemented using a dataset captured from students' interactions with a game-based learning environment designed to teach genetics. By generating features from the students' gameplay trace data, we are able to induce student competency models without utilizing inherently intrusive or distracting methods such as external assessments or data capture through physical sensors. Student competencies for the different genetics concepts presented in *Geniventure* are quantified using a post-test knowledge assessment, with different questions corresponding to different concepts within the game's individual levels. These levels are divided into "seen" and "unseen" groups for evaluation of the ZSL data augmentation performance.

3.1 *Geniventure* Learning Environment

To evaluate the impact of our generative ZSL approach to student competency modeling, we use gameplay interaction log data captured from students engaging with *Geniventure*. *Geniventure* is targeted towards middle and high school students (ages 11-18 years) and the overall design of the game is guided by fundamental genetics concepts aligning with the Next Generation Science Standards [30]. In *Geniventure*, students are faced with the challenge of correctly breeding and studying virtual drakes, a model species of dragon [13]. In order to successfully produce the desired drake for each in-game exercise, students are required to learn and explore

genetics concepts such as heredity, dominant and recessive traits, and protein-to-trait relationships.

The game is comprised of 60 progressively difficult puzzle-like challenges that are divided into six distinct levels (Figure 1). Each level is divided into different in-game "missions", and each mission is comprised of the individual challenges. The types of challenges presented to the student varies widely across the different levels. *Geniventure* is designed to be played in a linear sequence, but students are allowed to attempt any challenge at any time, and also have the freedom to quit any challenge prior to completion.

For the first three levels of *Geniventure*, students are faced with the task of modifying the genotype of a presented drake to match a target phenotype (Figure 1, Level 1). These challenges require the student to understand several different genetics concepts and also correctly predict attributes of the target phenotype based on the current genotype. To determine whether the student correctly completed the challenge or not, the student selects the "Check" or "Hatch" button in the game's interface to submit their answer. At this point, the student receives binary feedback on whether the exercise was successfully completed or not. If the exercise was not successfully completed at this time, the student is presented with a hint to help reach completion. There are three hints available per exercise, with each hint becoming progressively more direct, a visual cue is also made available to the student at this time. The student is allowed to make additional changes and resubmit their proposed phenotype until the correct solution is reached, or until the student exits the exercise. The progressive hint mechanism exists for many exercises in *Geniventure*, although the structure and conceptual challenges in each exercise may vary widely. Level 2 further advances the challenge of matching a genotype to a specified phenotype by introducing dominant and recessive traits (Figure 1, Level 2), while Level 3 increases the complexity of the exercise by adding factors such as scale color, proteins, and cell modification (Figure 1, Level 3). Levels 4, 5, and 6 introduce increasingly complex concepts such as breeding, inheritance, and meiosis. Level 4 introduces more complexity to the breeding process through the use of gametes, epistasis, and more challenging inheritance patterns (Figure 1, Level 4). Level 5 presents students with the test cross concept, a genetic method for determine an organism's genotype

by crossing it with a fully recessive organism (Figure 1, Level 5). Level 6 builds on the prior concepts of the game while introducing additional concepts such as X-linked and polyallelic traits (Figure 1, Level 6). Certain levels introduce gameplay narratives or scaffolding that are not present in prior levels. For example, Level 3 uses a “pod-release gate” interface which varies from Level 4’s “Gamete Builder”, Level 5’s “Test Cross” interface, and Level 6’s “Clutch Breeder”. As a result, this may result in different problem-solving behavior distributions while labels for the new concepts are not available yet to train competency models and thus provides the motivation for investigating model generalization techniques such as domain adaptation or zero-shot learning.

3.2 Data Collection

Following Institutional Review Board (IRB) approval, the data corpus was captured from 462 consenting students across seven high schools and a middle school located in the Eastern United States from suburban, rural, and urban locations. Teachers led several different classroom implementations of *Geniventure* where students engaged with the game during instruction periods across multiple days. Prior to the first learning session, students took a pre-test knowledge assessment consisting of 28 questions related to the genetics concepts presented in the game. Following the conclusion of the last learning sessions, students took an identical post-test knowledge assessment addressing the same concepts to quantify students’ learning gain. The knowledge assessment was aligned with *Geniventure*’s competency model based on the ECD formulation of the game, and the assessment was also validated through multiple rounds of expert review and cognitive interviews with students. The knowledge assessment’s administration showed an internal consistency reliability of $\alpha=0.873$, and both the pre- and post-test were administered through the same online platform as the game. Logistical and technical issues were encountered during the data collections, which resulted in 38 students being removed from the data corpus due to missing knowledge assessment data and 108 students being removed due to missing gameplay log data. As a result, the data corpus is comprised of gameplay log data from 316 students. Student performance on the post-test ($M=19.33$, $SD=6.131$) was shown to be a statistically significant improvement over the pre-test ($M=14.41$, $SD=5.826$) according to a paired t -test ($t(316) = 14.663$, $p < .01$, Cohen’s $d = 0.823$). The distribution of completed challenges per student appears to be relatively normal (Figure 2), with most students completing between 50-150 challenges, with a range of 5 to 248 challenges ($M=95.89$, $SD=33.63$).

3.3 Feature Engineering

Features representative of students’ gameplay were engineered from the raw timestamped log files generated for each learning session. These log files contained action-level information about students’ in-game activity, such as moves made within a challenge, number of attempts, and number of hints received. Because of the differences in gameplay mechanics across the different levels and challenges, we generated nine generic, challenge-level representations of student activity to generalize the feature engineering process. The generic representations were: (1) level number of challenge, (2) mission number of challenge, (3) challenge number, (4) total time spent on challenge, (5) number of in-game actions taken during challenge, (6) number of hints encountered during challenge, (7) number of correct in-game actions taken during challenge, (8) number of wrong in-game actions taken during challenge, and (9) student’s completion status of challenge (0: incomplete, 1: complete with wrong answer, 2: complete with correct answer). From these representations, we also generate additional features to capture the temporal context of students’ activity across

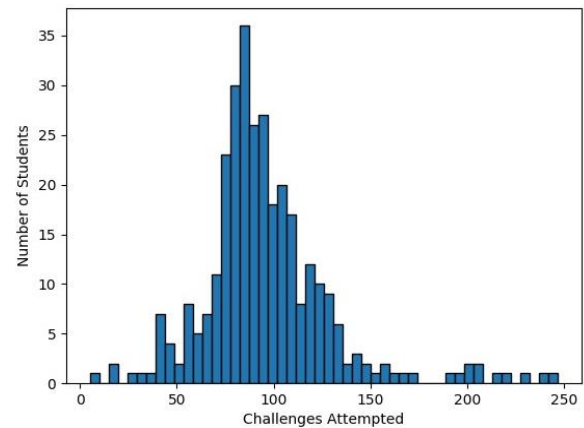


Figure 2. Histogram of challenges attempted by students in *Geniventure*.

current and prior learning sessions. By accounting for all student challenges completed up until the current challenge, we calculated: (10) average time spent on individual challenges (seconds), (11) average in-game movements per challenge, (12) average correct in-game movements per challenge, (13) average incorrect in-game movements per challenge, (14) average number of hints received per challenge, (15) successful challenge completion rate, (16) failed challenge completion rate, and (17) unsubmitted challenge rate. For each challenge, a feature vector was constructed using these 17 features to serve as the input to the student competency models. It should be noted that these averages were computed separately across the “seen” and “unseen” gameplay data to protect against data leakage.

4. EVIDENCE-CENTERED DESIGN

Stealth assessment refers to non-intrusive methods for collecting evidence to induce student competency models [38]. This evidence captured from student interactions with various learning platforms is subsequently used to inform the evidence models, and subsequently competency models. The competency predictions generated from these models can be used to enable enhanced learning through adaptive mechanisms within the learning platforms. Stealth assessment can also be used to inform teachers and instructors in real time about students’ current learning trajectories to determine if dynamic interventions are necessary.

Stealth assessment is grounded in Evidence-Centered Design (ECD), a principled approach to assessment design that describes high-level models of a conceptual assessment framework and delivery architecture for assessment delivery systems [27]. ECD affords assessment designers a method for reasoning about student knowledge or skills while adhering to psychometric principles [28]. Historically, ECD has been utilized to guide the creation of various knowledge assessments, and more recently has been used to inform the development of stealth assessment models deployed within game-based learning environments [14, 24, 37]. Our approach to stealth assessment with the *Geniventure* learning environment is informed by the three following ECD models:

Task Model: This model defines the exercises or activities that students attempt to complete through interactions with the game-based learning environment. The task model within *Geniventure* is comprised of sixty distinct challenges that are split across the six in-game levels. Each challenge presents students with various genetics concepts such as inheritance patterns, breeding, and genotype-phenotype relationships.

Evidence Model: This model is shaped by the actions performed by each student within the game-based learning environment. This interaction data is representative of student behavior that is correlated with learning outcomes pertaining to specific concepts presented in the learning platform. In this work, the actions a student performs in *Geniventure* challenges, in addition to the outcome of each challenge, are represented by the evidence model and used to engineer features to train machine learning models used for predicting student mastery (or no mastery) of particular genetics concepts. The evidence model guides the competency model as it adjusts its modeling of students’ competencies as various in-game challenges are attempted.

Competency Model: This model pertains to the genetics concepts that are presented within *Geniventure* and attempts to model the machine-interpretable evidence from the evidence model in order to accurately predict students’ competencies for each concept. The primary objective of the competency modeling is to optimally map the evidence model to the competency model for each student. These concepts are derived from classroom learning objectives and state science standards through of expert review. Students’ competencies are captured and quantified from a post-test knowledge assessment administered to each student following interactions with *Geniventure*. The competency model and the post-test assessment are aligned using the same concepts presented in Table 1.

To generate the ground-truth competency scores for each student, individual responses to the items on the post-test assessment are summed across the different genetics concepts, with a single concept mapping to between one and six questions on the assessment. Each question considered in this study is graded as either 1 (correct) or 0 (incorrect). Competency scores are calculated for each concept by dividing the total number of correct responses for that concept by the total number of questions related to that concept. As a result, each competency score was within the range of 0 to 1 and serves as the target variables for the stealth assessment models.

5. ZERO-SHOT LEARNING

Zero shot learning (ZSL) is an extreme variation of unsupervised domain adaptation, which focuses on two distinct data sets extracted from two different domains and data distributions: a *source domain* and a *target domain*. The primary objective of unsupervised domain adaptation is to induce a generalizable model that is capable of accurately classifying samples from the two domains in instances where labels for samples in the target domain are not available during model training. However, ZSL expands on this concept by assuming that neither data samples nor labels for the target domain are available during the training process. For this work, we split the *Geniventure* dataset into seen (S) and unseen (U) domains by partitioning between in-game levels because different levels address concepts that correspond to different competencies according to the ECD model. The text descriptions of each ECD concept serve as the link between the seen and the unseen classes, commonly referred to in ZSL as “semantic embeddings” or “attribute vectors” and are known for both seen and unseen classes at training time. The seen domain serves as the “source” domain and the unseen domain serves as the “target” domain. Therefore, the ECD concepts (C) for this data corpus (X : data, Y : class labels) can be divided and formally defined as follows:

$$S = \{(x, y, c_y) \mid x \in X^S, y \in Y^S, c \in C^S\} \quad (1)$$

$$U = \{(x, y, c_y) \mid x \in X^U, y \in Y^U, c \in C^U\} \quad (2)$$

By framing the ECD competency modeling as a ZSL task, we seek to predict student competencies on unseen levels and concepts in situations where no prior gameplay data is available for those concepts, an example of the “cold-start problem”. For example, if we have concepts C1-C10 where C1-C7 have been previously presented to students in *Geniventure* and gameplay logs have been captured for these concepts (i.e., seen), we seek to use this available data to induce generalizable competency models that accurately predict student outcomes on concepts C8-C10, even though they

Table 1. In-game genetics concepts from ECD competency model.

Concept	Concept Description	Questions
C1	Only one dominant allele is needed to produce the dominant trait.	3
C2	Two recessive alleles are needed to produce a recessive trait.	2
C3	Create or select parental gametes to create an individual offspring with a specific phenotype.	4
C4	Set parental genotypes to produce a specific pattern of offspring.	6
C5	Use patterns in the phenotypes of a group of offspring to predict the genotype of the parents.	5
C6	For some traits primarily influenced by a single gene, both alleles will have some effect, with neither being completely dominant.	2
C7	Breed with a recessive animal to determine an unknown genotype (testcross).	2
C8	Different versions of a gene correspond to different versions of a specific protein.	2
C9	Proteins do work or have jobs to do in cells.	1
C10	Proteins are nanomachines; different proteins do different jobs.	1
C11	The function of a protein is determined by its shape.	1
C12	Different versions of a specific protein have different structures and different functions.	1
C13	Some traits have multiple alleles, which can form a ranked series in terms of dominance.	2
C14	Genes on the X chromosome are referred to as X-linked. Males receive only one copy of the X chromosome and pass on their X only to their daughters.	1
C15	Working from the phenotype, determine possible genotypes for an organism.	2
C16	Use a genotype to predict the phenotype for an organism.	2

have not been presented to any students and there is no prior gameplay logs or competency scores for these concepts (i.e., unseen). This allows for more generalizable student competency modeling and also enables ECD-based stealth assessment in circumstances where there are no prior ground-truth competency labels, such as when post-tests may be unavailable or impractical to administer. Examples include informal learning environments such as museums, or if a concept or level within a game-based learning environment is being deployed for the first time.

Because both the data and labels are not available at training time, a form of semantic data must be available for the ZSL framework to link between the seen and unseen domains. In this particular case, we use the text-based descriptions of each of the concepts to generate concept mastery embeddings used as conditional inputs to the generative ZSL models (Section 5.1). This allows the generative model to learn the non-linear relationships between the concept text embeddings, student competencies, and student gameplay patterns. As a result, the generative model generates synthetic gameplay data representative of each of the seen and unseen concepts which enhances the training and generalizability of the competency models.

The embedding representations for the genetics concept descriptions are generated using Sentence-BERT (S-BERT) [33]. S-BERT expands upon the original BERT model [10] by implementing the BERT model within a Siamese network architecture to facilitate the generation of fixed-length embedding vectors of sentences that are compared using distance metrics such as cosine similarity. This allows the S-BERT model to generate sentence-level embedding representations instead of single-word embeddings, making this model suitable for sentence descriptions of the genetics concepts. Additionally, the use of a common pre-trained language model such as S-BERT improves the generalizability of our approach, compared to manually crafted representations such as knowledge graphs.

5.1 Generative Modeling

To address the absence of training data and labels for the unseen concepts, we use semantic data (i.e., text descriptions of all concepts) to condition and train deep generative models, which facilitate data augmentation for this task. By employing text embeddings of the seen concept descriptions to condition the generative models, we can train the models to map the latent representation of each concept to particular patterns and features in the students’ gameplay data for the seen concepts. We can subsequently generate synthetic data to represent student gameplay for the unseen concepts using only the embeddings of the descriptions for each unseen concept. The augmented data is used to further train the competency models to increase the predictive performance during inference for the unseen concepts’ associated gameplay data.

Generative adversarial networks (GANs) have been frequently used as a data augmentation method due to their ability to generate high-fidelity data from noise vectors through zero-sum training of two deep learning components: a *generator* and a *discriminator* [15]. The purpose of the generator G is to generate synthetic data \tilde{x} based on a random probability distribution p_z , where z represents the latent space sampled by G so that $\tilde{x} = G(z)$, $z \sim p_z$. The objective is that \tilde{x} deceives the discriminator, whose purpose is to accurately distinguish between this “fake” data and real samples from the original data. The discriminator’s training loss from this binary classification is backpropagated through the generator and the discriminator, with the objective of both losses eventually reaching a Nash equilibrium. However, quantifying GAN convergence is an open-ended area of research and GAN models are often susceptible

to vanishing or exploding gradients, mode collapse, and other instabilities during the training process. One approach to mitigating this issue is a conditional GAN [25], which extends a traditional GAN architecture by providing associated data (“conditions”) to both the generator and discriminator’s input vectors. An example of such a condition is a class label or attribute that is associated with the desired augmented output of the generator. For this particular case, we use the S-BERT embedding vectors of the associated concepts as the conditions to our GAN model to generate synthetic gameplay data associated with both the seen and unseen concepts.

Traditional GAN architectures attempt to reach convergence by minimizing a divergence function such as the Jensen-Shannon (JS) divergence, which helps quantify the distances between two probability distributions $p_g(x)$ and $p_r(x)$, where p_g is the model distribution of the generator and p_r is the distribution of the real data. However, a common issue with these divergence metrics is that there exist sequences of distributions that do not converge under the JS divergence or where the gradient of the divergency eventually disappears, effectively halting the training of the generator during backpropagation. To address this issue, an alternative GAN architecture (W-GAN) was proposed that utilizes the Wasserstein distance, otherwise known as the “Earth Mover’s distance”, as a means to quantify the generator loss during training (Eq. 3) [2]. This metric is desirable as it is continuous and differential almost everywhere under the Lipschitz condition:

$$W(p_r, p_g) = \inf_{\gamma \in \Pi(p_r, p_g)} \mathbb{E}_{(x,y) \sim \gamma} [\|x - y\|] \quad (3)$$

where $\Pi(p_r, p_g)$ represents all joint distributions with marginal distributions of $\gamma(x, y)$ are $p_r(x)$ and $p_g(y)$, respectively. Because this equation is highly intractable, we use the Kantorovich-Rubinstein duality to simplify the calculation to be:

$$W(p_r, p_g) = \sup_{|f|_{L^1} \leq 1} \mathbb{E}_{x \sim p_r} [f(x)] - \mathbb{E}_{x \sim p_g} [f(x)] \quad (4)$$

As a result, we enforce a 1-Lipschitz constraint to the discriminator component. As a means to enforce this 1-Lipschitz constraint within the W-GAN, we introduce a concept known as “weight clipping” to the discriminator. This involves constraining the weights in the discriminator to be between the range of $[-c, c]$, with c being treated as an additional training hyperparameter. However, weight clipping is often volatile with respect to c and can cause W-GANs to converge much more slowly if c is too large but can also introduce vanishing gradients if c is too small. An alternative to the weight clipping is a “gradient penalization” method which proposes a penalty term added to the loss function that is parameterized by a penalty coefficient λ [17]. The gradient penalty term is based on weighted random sampling between the real and the generated samples from the generator. As a result, the final objective of our gradient-penalized W-GAN model (WGAN-GP) becomes the minimization of the following loss function for the discriminator D :

$$\mathcal{L}_{dis}(x, \tilde{x}; \theta_{dis}) = D_{\theta_{dis}}(\tilde{x}) - D_{\theta_{dis}}(x) + \lambda(\|\nabla_{\tilde{x}} D_{\theta_{dis}}(\tilde{x})\| - 1)^2 \quad (5)$$

where θ_{dis} represents the parameters of the discriminator, λ is the gradient penalty coefficient, and \tilde{x} is sampled from $\epsilon x + (1 - \epsilon)\tilde{x}$ with $0 \leq \epsilon \leq 1$, effectively representing any points sampled between the probability distributions, p_g and p_r .

We employ the WGAN-GP approach as a means to train a generator to produce realistic synthetic data representing students’ gameplay for unseen *Geniventure* levels in order to induce student competency models for the associated genetics concepts that have

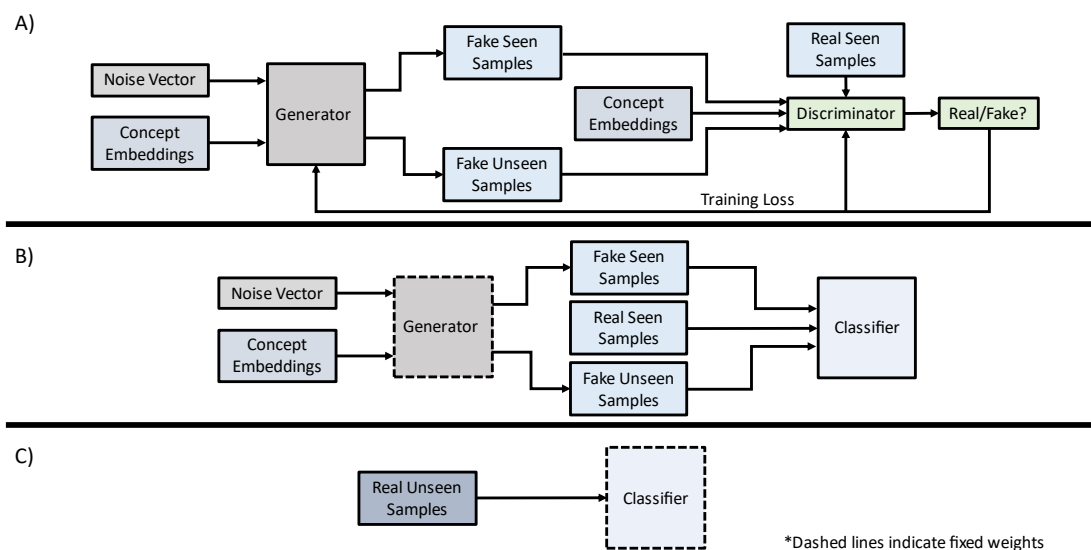


Figure 3. Visualization of data augmentation in generative zero-shot learning framework. (A) training process for conditional WGAN-GP augmentation model, (B) training process for student competency models with augmented data for unseen concepts, and (C) inference of trained competency model for real student gameplay on previously unseen concepts.

not been previously introduced. This model was selected due to issues with vanishing gradients during empirical evaluations of traditional GANs and WGANs within our ZSL framework. Additionally, we use a conditional variation of the WGAN-GP model in order to effectively map the latent representations of the augmented data representations to the word embeddings of the text-based descriptions of each genetics concept. This allows the synthetic data generation to be guided by the concepts associated with each data sample from the data corpus, while allowing for data to be generated based on the concepts associated with the unseen game levels.

6. METHODOLOGY

Our ZSL approach is evaluated across two different data splits. The first split involves removing any information from the ECD models associated with Level 6 (the last level) from the data corpus, which includes student gameplay (Evidence Model) from the challenges within the level (Task Model), as well as the student post-test scores (Competency Model) to items associated with genetics concepts associated with Level 6 (C13 and C14 serving as unseen concepts). The second split involves treating Level 5 and Level 6 as the “unseen” domain, which involves concepts C7, C13, and C14. The competency models are initially trained using the student gameplay features from the “seen” domain and a median split of the combined competency scores from the post-test items corresponding only to concepts associated with the “seen” *Geniventure* levels. To evaluate the ZSL performance, each competency model is evaluated using the gameplay from the unseen levels to predict student competencies from the post-survey items associated with the unseen concepts. This process is implemented to protect against data leakage, ensuring that the student competency models are induced using only data and labels from the seen levels as well as unseen, synthetic labeled data during the training phase, and any actual data or labels from the unseen concepts and levels are *only* presented to each competency model during the inference phase (Figure 3, C).

6.1 Student Competency Models

Five different classifiers were investigated as student competency models: a majority classifier, support vector machine (SVM), random forest (RF), Naïve Bayes (NB), logistic regression (LR), and

feedforward neural network (FFNN). Each of the models were implemented as binary classifiers to predict “high” and “low” categories of student performance based on a median split of the sum of the post-test score for each question related to either the seen or unseen concepts. Each of the models was evaluated using student-level cross-validation, to ensure against data leakage across validation folds. The median of the competency scores were determined based on the scores of the students within the training folds only as another means to protect against data leakage.

6.2 Model Evaluation

Hyperparameter tuning was performed using three-fold inner cross-validation within each iteration of the ten-fold outer cross-validation. The hyperparameters that were optimized were the regularization parameter and kernel (support vector machine; SVM), regularization parameter (logistic regression, LR), number of estimators (random forest; RF), and number of layers and nodes (feedforward neural network; FFNN). Hyperparameter tuning was not performed on the Naïve Bayes classifier and the majority classifier. Because multiple data samples exist per student and there is only one competency label per student, we generate a single student-level prediction by forward propagating all feature vectors for a given student through a trained competency model and averaging across all predictions for that student.

Because the use of the gradient penalty in the WGAN-GP stabilizes the training process and mitigates the need for extensive hyperparameter tuning, we focus only on tuning the number of nodes in the two hidden layers of the generator and discriminator using layer sizes of 32 or 64. This hyperparameter tuning also occurred within the nested cross-validation process described previously. The WGAN-GP’s learning rate was 0.001, with a dropout rate of 0.5 and hyperbolic tangential activation functions. The WGAN-GP (and FFNN competency model) was trained using 100 epochs while utilizing early stopping based on the model’s performance on the validation fold with a patience of 10 epochs. The noise vector size for the WGAN-GP was 32, and the number of generated synthetic data samples was set to be 50% of the original training dataset size. Because the concept mastery embedding vectors obtained

from S-BERT are high in dimensionality, we perform Principal Component Analysis on the embeddings to reduce the size to 32 components. The data was standardized within each cross-validation fold by subtracting each feature’s mean and dividing by the standard deviation of each feature as determined by the training folds.

To represent the students’ competencies within the semantic embeddings, a text description for each concept was preceded with either “mastery of” or “no mastery of” based on each student’s post-test performance relative to the median for each concept and these substrings were concatenated together to form a single comprehensive text string representing the student’s mastery of each of the seen concepts. For example, a concept mastery sentence for a single student might be “Mastery of *only one dominant allele is needed to produce the dominant trait*. Non-mastery of *two recessive alleles are needed to produce a recessive trait*. Mastery of ...” and so on for the seen concepts. These text representations are then passed through the pre-trained S-BERT word embedding model and, following the PCA dimensionality reduction described previously, are used as the conditional features for the generative models. To generate synthetic data following the generative model training, a similar process is followed to generate the text representations using the descriptions of the unseen concepts. As no student competency data actually exists for these unseen concepts currently, the preceding phrase “mastery of” or “no mastery of” is determined using a Bernoulli probability distribution where probability $p=0.5$. This allows the generative model to produce synthetic gameplay data representative of many possible mastery/non-mastery combinations of student competencies for the unseen concepts, thus enhancing the generalizability of the competency models for the unseen concepts.

To evaluate the performance of the WGAN-GP as the preferred ZSL generative model, we also investigate two alternatives: a conditional variational autoencoder (C-VAE) and a “target-only” baseline. The target-only baseline refers to a competency model that performs the inference on the data for the unseen *Geniventure* levels but does not undergo any form of ZSL-based data augmentation, which is a reflection of the target-only baseline in prior adversarial domain adaptation work by Tzeng et al. [43]. C-VAEs are similar to conditional GANs with regards to the use of conditional attributes [39]. A traditional VAE contains two components: an *encoder* and a *decoder*. The encoder learns latent representations of input data while the decoder seeks to reconstruct the original dimensionality of the data from the latent space representation. The VAE constrains the latent space representation to follow a pre-determined probability distribution by minimizing a loss function that consists of a reconstruction term and a divergence term. The reconstruction term quantifies the reconstruction error of the decoder component through a loss function such as root mean squared error and the divergence term quantifies the distance between the given probability distribution and the latent representation distribution. The Kullback-Leibler (KL) divergence is often used for this purpose. The conditional features are concatenated to the input features for the encoder as well as the latent representation vector that is passed from the encoder to the decoder. Because the encoder reduces the latent representation to a parameterized probability distribution, this allows the decoder to generate augmented data from this distribution.

7. RESULTS AND DISCUSSION

The ZSL approach was evaluated across two splits as described in Section 5. The results for Split 1 is shown in Table 2, while the

results for Split 2 is shown in Table 3. We select F1 Score and accuracy as our primary metrics to account for the relatively balanced class distribution due to the median split, while Area-Under-Curve (AUC) and Cohen’s Kappa [8] are used as secondary metrics. The optimally performing generative ZSL model for each split in terms of F1 Score is shown in bold. All evaluations were performed on a NVIDIA GeForce GTX 1080 TI GPU. Each evaluation took up to 100 minutes to complete the 10-fold cross-validation sequence.

In terms of the primary evaluation metrics, the WGAN-GP model appeared to induce the highest performance from the competency models across both data splits, outperforming both the C-VAE and the target-only baseline. It was noted that the performance across all models decreased for the split containing two “unseen” gameplay levels (Table 3) compared to one (Table 2), which is expected due to the decrease in training data available as well as the increased variance in the “unseen” dataset. However, the results overall point to the enhanced performance of the student competency models when additional synthetic data is generated from the WGAN-GP as a means to improve the training process.

Additionally, it was noticeable that the margin between the WGAN-GP and the other ZSL configurations widened from Split 1 to Split 2, which points to the relative scalability of our approach as the number of unseen concepts and in-game levels increase. It was also noted that the C-VAE was the lowest performing generative model and was also outperformed by the target-only baseline approach for both data splits. This is noteworthy as VAEs, including conditional variations, are the generative approach for prior generalized ZSL work [26, 44, 45].

However, we note in these prior works, the VAE models were trained on a multimodal dataset, which provides a more data-rich perspective compared to the stealth assessment data in this work. Additionally, although the work in [26] used a conditional variation of the C-VAE, the generative model appeared to suffer from mode collapse, a common issue in the training of generative models. However, one benefit of the WGAN (and the gradient-penalization modification) is additional mitigation against mode collapse during training, a possible explanation of why the WGAN-GP achieves the highest performance in our evaluations. A primary difference in this architecture is that the loss of the decoder in the C-VAE is the summation of the KL divergence and the reconstruction loss, compared to the WGAN component which strictly uses the Wasserstein divergence metric. This has potential for allowing the generative model to map between the semantic feature space and the augmented data more effectively, particularly as the augmented data from the WGAN-GP is restored to the original dimensionality instead of a latent space representation.

To further investigate the performance of the generative ZSL approach, we generate the confusion matrices for each of the models across both data splits (Figures 4 and 5). The confusion matrices are based on the inferences of each of the models based on the held-out test set within each outer cross-validation iteration, for a total of 316 student-level predictions. It should be noted that the results in Tables 2 and 3 were calculated across the outer cross-validation folds while the analyses conducted in Figures 4-7 were calculated across the entire dataset. Based on the results in the confusion matrices, we observe that high-performing students are more accurately classified compared to low-performing students. Additionally, it appears that the student competency models produced noticeably more false negatives as the unseen concepts increased. This occurred across all three evaluated ZSL approaches. In the case of the target-only baseline and the WGAN-GP model, the student competency models were able to retain a relatively similar

Table 2. Results of ZSL framework for Split 1.

ZSL Model	Classifier	F1 Score	Accuracy	AUC	Kappa
Target-Only	SVM	0.689	0.642	0.689	0.257
CVAE	RF	0.668	0.642	0.675	0.275
WGAN-GP	SVM	0.709	0.656	0.692	0.284

Table 3. Results of ZSL framework for Split 2.

ZSL Model	Classifier	F1 Score	Accuracy	AUC	Kappa
Target-Only	SVM	0.671	0.623	0.703	0.249
CVAE	FFNN	0.612	0.578	0.700	0.152
WGAN-GP	SVM	0.696	0.652	0.696	0.281

performance for correct identification of low-performing students across both data splits, but the C-VAE led to significantly decreased detection for low-performing students when evaluating from Split 1 to Split 2. Additionally, it is noticeable that the WGAN-GP was able to maintain relatively consistent performance for prediction of both high-performing and low-performing students across both data splits, which demonstrates the generalizability of this particular generative model.

To further evaluate the predictive value of the semantic embeddings of the student competencies with the various genetics concepts, we visualize the embeddings from each of the students using the principal components generated from the S-BERT embeddings. Using the t-distributed Stochastic Neighbor Embedding (t-SNE) plots for the low-performing and high-performing students (Figure 6), we are able to detect whether there are salient or underlying predictive patterns in the text representations of each student’s mastery of individual genetics concepts. Despite the use of PCA for dimensionality reduction for the original S-BERT embeddings, the representations of the semantic embeddings contain high dimensionality and thus poses a challenge for visualization. t-SNE attempts to address this issue by producing a representation of high-dimensionality data within 2D coordinate space. This is performed by constructing joint probability distributions to model the similarity between the original data points, and subsequently attempts to

minimize the KL divergence between these probability distributions and other probability distributions within 2D coordinate space. This process is expanded to distinguish between low-performing and high-performing students for the seen and unseen concepts (Figure 7). Figures 6 and 7 are generated using Split 2.

Figure 6 indicates that the use of semantic embedding representation of the students’ masteries of various genetics concepts may provide predictive context to the generative ZSL models when used as a conditioning input during training. Noticeably, there appears to be distinct separation between the semantic embeddings for high-performing students and low-performing students when calculated using all competencies, which points to the predictive value of using these embeddings for stealth assessment tasks. To provide analysis more similar to the ZSL framework, the semantic embeddings are generated for each group of low-performing and high-performing students by separating the embeddings for seen and unseen concepts (Figure 7). In this particular case, there appears to be notable separation between clusters of low-performing students and high-performing students, with overlap between the high-performing students across both seen and unseen concepts. As a result, this indicates that the use of the semantic embeddings alongside conditional generative modeling provides additional predictive value to guide the generation of augmented data for different students based on prior competencies. One aspect of note for Figure 7 is that the

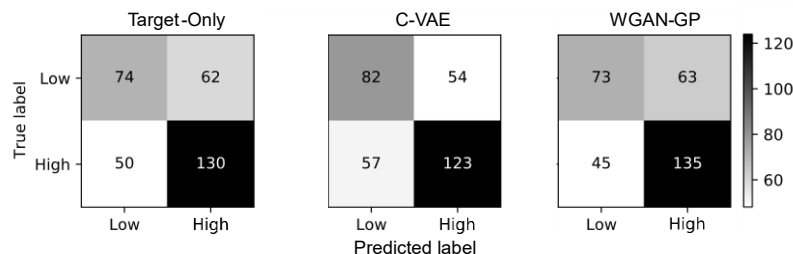


Figure 4. Confusion matrices for baseline and generative ZSL models (Split 1).

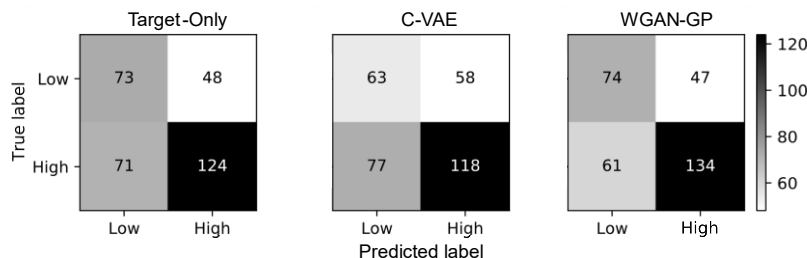


Figure 5. Confusion matrices for baseline and generative ZSL models (Split 2).

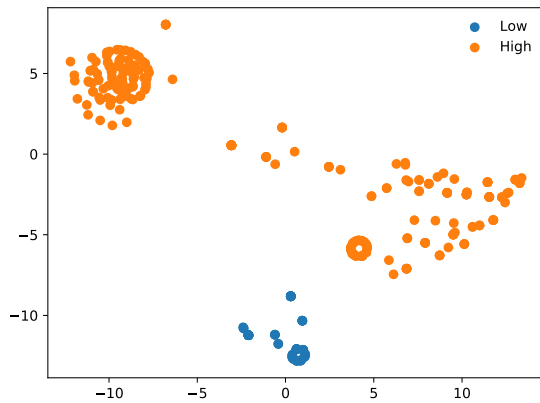


Figure 6. t-SNE visualization of student competency S-BERT embeddings across all concepts.

plots of “High Unseen” and “Low Unseen” are based on the ground-truth competencies for the students on the unseen concepts, while in practice, this data is not available for training the stealth assessment models and the semantic representations are used by assigning “high” or “low” mastery of each concept at random and then generating a synthetic binary “label” based on whether at least 50% of the unseen concepts were labeled “high” or not. This allows the generative model to be conditioned on 2^n different combinations of student concept mastery where n is the total number of unseen concepts, and this allows the model to be trained using a higher number of mastery combinations than what is often available in datasets captured from game-based learning environments.

There are limitations to this work that should be noted. Although the zero-shot learning framework was based on seen and unseen domains across differing gameplay levels, the two domains were grounded in the same game-based learning environment. To further investigate the generalizability of the ZSL framework, our approach should be evaluated using unseen data and classes from entirely different learning environments. Additionally, Split 1 and 2 removed two and three concepts out of sixteen, resulting in 12-18% of the total data being treated as unseen data. Evaluations with more unseen in-game levels would provide more insight into the performance of our approach as the unseen domain increases. The class label for the seen and unseen domains were both based on binary labels of student mastery, but our method should also be evaluated in scenarios where the labels differ more widely (e.g., an additional unseen class in multi-class prediction). The binary labels

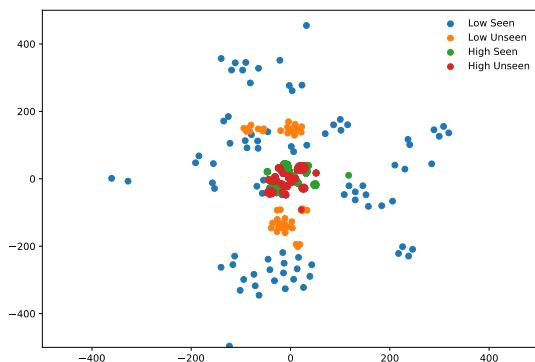


Figure 7. t-SNE visualization of student competency S-BERT embeddings across high/low and seen/unseen splits.

were utilized to convert our work to a classification task, but the level of granularity is much higher compared to regression, which

can negatively impact the adaptability of learner-sensitive mechanisms by grouping low-performing students and students performing near the median together. It was also noted that, as the unseen domain increased, the number of false negatives increased, which could lead to high-performing students receiving unnecessary interventions in user-adaptive settings.

8. CONCLUSION

Game-based learning holds significant potential for stealth assessment of student performance and knowledge acquisition. The capability to predict student mastery of particular concepts within game-based learning environments can enable mechanisms such as adaptive hint generation, personalized gameplay narratives and scaffolding, and gameplay-sensitive interventions in real-time. However, stealth assessment models often necessitate large amounts of data and labels, which presents logistical and scalability challenges. This prohibits the deployment of pre-trained stealth assessment models in domains where prior data and labels have not been collected, and questions remain regarding generalizability to different domains and educational content.

We propose a generative zero-shot learning framework to address the above issues. By using conditional generative models, we harness the predictive capabilities of textual representations of student mastery of different educational concepts. These representations are able to guide a Wasserstein Generative Adversarial Network in generating synthetic student gameplay data representative of in-game levels and genetics concepts that have not been previously presented and for which no prior gameplay data or student competency data actually exists. By mapping text embeddings of genetics concepts to the student gameplay data through the generative model, the resulting augmented data improves the predictive capacity of stealth assessment models for predicting student competency across different hidden gameplay levels. Our proposed model is shown to outperform an alternative conditional generative model and a baseline that excludes the zero-shot learning element. This indicates the potential for increasing the generalizability of student stealth assessment models through the generative data augmentation approach and for deploying pre-trained stealth assessment models in digital learning environments presenting new educational concepts, problem-solving tasks, and in-game levels.

There are many promising avenues for future work. Notably, our work focuses on zero-shot learning within a single game-based learning environment, and the natural extension of this work is the evaluation of our framework across different learning environments instead of separate in-game levels. Additional experimentation with a higher ratio of “unseen”-to-“seen” concepts would provide more insight into how the ZSL framework’s performance is maintained as the amount of “unseen” data increases in size and variance. More complex modeling for the stealth assessment, language embeddings, and generative models may provide additional benefit for the predictive capacity of our framework. Finally, the effectiveness of our approach should be implemented alongside student-adaptive interventions to determine the impact on learning outcomes and processes within run-time environments.

9. ACKNOWLEDGMENTS

The authors would like to thank Robert Taylor for his assistance in facilitating this research. This research was supported by the National Science Foundation under Grant DRL-1503311. Any opinions, findings, and conclusions expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

10. REFERENCES

- [1] Akram, B., Min, W., Wiebe, E., Mott, B., Boyer, K. and Lester, J. 2018. Improving stealth assessment in game-based learning with LSTM-based analytics. In *Proceedings of the 11th International Conference on Educational Data Mining*, 208–218.
- [2] Arjovsky, M., Chintala, S. and Bottou, L. 2017. Wasserstein GAN. *arXiv:1701.07875*.
- [3] Asbell-Clarke, J., Rowe, E. and Terc, E. 2013. Working through impulse: Assessment of emergent learning in a physics game. *Games + Learning + Society*. 9, 1, 1–7.
- [4] Binkley, M., Erstad, O., Herman, J., Raizen, S., Ripley, M., Miller-Ricci, M. and Rumble, M. 2012. Defining Twenty-First Century Skills. *Assessment and Teaching of 21st Century Skills*.
- [5] Chaudhry, R., Singh, H., Dogga, P. and Saini, S. 2018. Modeling hint-taking behavior and knowledge state of students with multi-task learning. In *Proceedings of The 11th International Conference on Educational Data Mining*, 21–31.
- [6] Cheng, M.-T., Rosenheck, L., Lin, C.-Y. and Klopfer, E. 2017. Analyzing gameplay data to inform feedback loops in The Radix Endeavor. *Computers & Education*. 111, 60–73.
- [7] Clark, D., Tanner-Smith, E. and Killingsworth, S. 2016. Digital games, design, and learning: A systematic review and meta-analysis. *Review of Educational Research*. 86, 1, 79–122.
- [8] Cohen, J. 1960. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*. 20, 1, 37–46.
- [9] Delacruz, G., Chung, G. and Baker, E. 2010. Validity Evidence for Games as Assessment Environments. National Center for Research on Evaluation, Standards, and Student Testing.
- [10] Devlin, J., Chang, M.-W., Lee, K. and Toutanova, K. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv:1810.04805*.
- [11] Efremov, A., Ghosh, A. and Singla, A. 2020. Zero-shot learning of hint policy via reinforcement learning and program synthesis. In *Proceedings of The 13th International Conference on Educational Data Mining*, 338–394.
- [12] Falakmasir, M., Gonzalez-Brenes, J., Gordon, G. and DiCerbo, K. 2016. A data-driven approach for inferring student proficiency from game activity logs. In *Proceedings of the Third ACM Conference on Learning @ Scale*, 341–349.
- [13] McElroy-Brown, K. and Reichsman, F. 2019. Genetics with dragons: Using an online learning environment to help students achieve a multilevel understanding of genetics. Retrieved from <http://concord.org>.
- [14] Georgiadis, K., van Lankveld, G., Bahreini, K. and Westera, W. 2021. On the robustness of stealth assessment. *IEEE Transactions on Games*. 13, 2, 180–192.
- [15] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y. 2014. Generative adversarial networks. In *Advances in neural information processing systems*. 27, 1, 1–9.
- [16] Greipl, S., Moeller, K. and Ninaus, M. 2020. Potential and limits of game-based learning. *International Journal of Technology Enhanced Learning*. 12, 4, 1–45.
- [17] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V. and Courville, A. 2017. Improved Training of Wasserstein GANs. *arXiv:1704.00028*.
- [18] Henderson, N., Kumaran, V., Min, W., Mott, B., Wu, Z., Boulden, D., Lord, T., Reichsman, F., Dorsey, C., Wiebe, E. and Lester, J. 2020. Enhancing student competency models for game-based learning with a hybrid stealth assessment framework. In *Proceedings of the 13th International Conference on Educational Data Mining*, 92–103.
- [19] Hsiao, H.-S., Chang, C.-S., Lin, C.-Y. and Hu, P.-M. 2014. Development of children’s creativity and manual skills within digital game-based learning environment. *Journal of Computer Assisted Learning*. 30, 4, 377–395.
- [20] Hutt, S., Ocumpaugh, J., Biswas, G. and Baker, R.S. 2021. Investigating SMART models of self-regulation and their impact on learning. In *Proceedings of The 14th International Conference on Educational Data Mining*, 580–587.
- [21] Jesus, Â.M. de and Silveira, I.F. 2019. A collaborative game-based learning framework to improve computational thinking skills. In *Proceedings of The 2019 International Conference on Virtual Reality and Visualization*, 161–166.
- [22] Junokas, M.J., Lindgren, R., Kang, J. and Morphew, J.W. 2018. Enhancing multimodal learning through personalized gesture recognition. *Journal of Computer Assisted Learning*. 34, 4, 350–357.
- [23] Larochelle, H., Erhan, D. and Bengio, Y. 2008. Zero-data learning of new tasks. In *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence*, 646–651.
- [24] Min, W., Frankosky, M., Mott, B., Rowe, J., Smith, A., Wiebe, E., Boyer, K. and Lester, J. 2020. DeepStealth: Game-based learning stealth assessment with deep neural networks. *IEEE Transactions on Learning Technologies*. 13, 2, 312–325.
- [25] Mirza, M. and Osindero, S. 2014. Conditional generative adversarial nets. *arXiv:1411.1784*.
- [26] Mishra, A., Reddy, S., Mittal, A. and Murthy, H. 2018. A generative model for zero shot learning using conditional variational autoencoders. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2188–2196.
- [27] Mislevy, R., Almond, R. and Lukas, J. 2003. A Brief Introduction to Evidence-Centered Design. *ETS Research Report Series*. 2003, 1, 1–29.
- [28] Mislevy, R., Behrens, J., DiCerbo, K. and Levy, R. 2012. Design and discovery in educational assessment: Evidence-centered design, psychometrics, and educational data mining. *Journal of Educational Data Mining*. 4, 1, 11–48.
- [29] Missaoui, S. and Maalel, A. 2021. Student’s profile modeling in an adaptive gamified learning environment. *Education and Information Technologies*. 26, 5, 6367–6381.
- [30] NGSS Lead States 2013. *Next Generation Science Standards: For States, By States*. The National Academies Press.
- [31] Nguyen, H., Hou, X. and Stamper, J. 2020. Moving beyond test scores: Analyzing the effectiveness of a digital learning game through learning analytics. In *Proceedings of The 13th International Conference on Educational Data Mining*, 487–495.
- [32] Oliveira, W., Isotani, S., Pastushenko, O., Hruška, T. and Hamari, J. 2021. Modeling students’ flow experience through data logs in gamified educational systems. In *Proceedings of the 2021 International Conference on Advanced Learning Technologies (ICALT)*, 97–101.
- [33] Reimers, N. and Gurevych, I. 2019. Sentence-BERT: Sentence embeddings using siamese BERT-networks. *arXiv:1908.10084*.
- [34] Sabourin, J., Rowe, J., Mott, B. and Lester, J. 2013. Considering alternate futures to classify off-task behavior as emotion self-regulation: A supervised learning approach. *Journal of Educational Data Mining*. 5, 1, 9–38.
- [35] Shute, V., Ke, F. and Wang, L. 2017. Assessment and Adaptation in Games. *Instructional Techniques to Facilitate*

- Learning and Motivation of Serious Games*. P. Wouters and H. van Oostendorp, eds. Springer International Publishing. 59–78.
- [36] Shute, V. and Rahimi, S. 2021. Stealth assessment of creativity in a physics video game. *Computers in Human Behavior*. 116, 1–13.
- [37] Shute, V., Rahimi, S., Smith, G., Ke, F., Almond, R., Dai, C., Kuba, R., Liu, Z., Yang, X. and Sun, C. 2021. Maximizing learning without sacrificing the fun: Stealth assessment, adaptivity and learning supports in educational games. *Journal of Computer Assisted Learning*. 37, 1, 127–141.
- [38] Shute, V. and Ventura, M. 2013. *Stealth Assessment: Measuring and Supporting Learning in Video Games*. The MIT Press.
- [39] Sohn, K., Lee, H. and Yan, X. 2015. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems*, 2241–2251.
- [40] Spaulding, S., Shen, J., Park, H. and Breazeal, C. 2021. Towards transferrable personalized student models in educational games. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, 1245–1253.
- [41] Talandron, M., Rodrigo, Ma. and Beck, J. 2017. Modeling the incubation effect among students playing an educational game for physics. In *Proceedings of the 18th International Conference on Artificial Intelligence in Education*, 371–380.
- [42] Taub, M., Azevedo, R., Bradbury, A., Millar, G. and Lester, J. 2018. Using sequence mining to reveal the efficiency in scientific reasoning during STEM learning with a game-based learning environment. *Learning and Instruction*. 54, 93–103.
- [43] Tzeng, E., Hoffman, J., Saenko, K. and Darrell, T. 2017. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 7167–7176.
- [44] Verma, V., Mishra, A., Pandey, A., Murthy, H. and Rai, P. 2021. Towards zero-shot learning with fewer seen class examples. In *Proceedings of the 2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2241–2251.
- [45] Wu, M., Mosse, M., Goodman, N. and Piech, C. 2019. Zero shot learning for code education: Rubric sampling with deep learning inference. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, 782–790.
- [46] Zhang, J., Das, R., Baker, R. and Scruggs, R. 2021. Knowledge tracing models’ predictive performance when a student starts a skill. In *Proceedings of The 14th International Conference on Educational Data Mining*, 625–629.

Generalisable Methods for Early Prediction in Interactive Simulations for Education

Jade Maï Cock
EPFL
jade.cock@epfl.ch

Mirko Marras
University of Cagliari
mirko.marras@acm.org

Christian Giang
EPFL
christian.giang@epfl.ch

Tanja Käser
EPFL
tanja.kaeser@epfl.ch

ABSTRACT

Interactive simulations allow students to discover the underlying principles of a scientific phenomenon through their own exploration. Unfortunately, students often struggle to learn effectively in these environments. Classifying students' interaction data in the simulations based on their expected performance has the potential to enable adaptive guidance and consequently improve students' learning. Previous research in this field has mainly focused on a-posteriori analyses or investigations limited to one specific predictive model and simulation. In this paper, we investigate the quality and generalisability of models for an early prediction of conceptual understanding based on clickstream data of students across interactive simulations. We first measure the students' conceptual understanding through their in-task performance. Then, we suggest a novel type of features that, starting from clickstream data, encodes both the state of the simulation and the action performed by the student. We finally propose to feed these features into GRU-based models, with and without attention, for prediction. Experiments on two different simulations and with two different populations show that our proposed models outperform shallow learning baselines and better generalise to different learning environments and populations. The inclusion of attention into the model increases interpretability in terms of effective inquiry. The source code is available on Github¹.

Keywords

Simulations, Time Series Classification, Early Prediction, Conceptual Understanding, GRU, Self-Attention

1. INTRODUCTION

In the past years, interactive simulations have gained increasing popularity in formal education [6] and have become integral parts of many science curricula [28]. These environments provide visualisations of abstract concepts, which can help students to better grasp them [34, 24]. On the other

¹<https://github.com/epfl-ml4ed/beerslaw-lab.git>

J. M. Cock, M. Marras, C. Giang, and T. Käser. Generalisable methods for early prediction in interactive simulations for education. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 183–194, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852968>

hand, these simulations also allow students to explore new learning content and build knowledge in an active and independent way [33]. Since simulations are virtual learning environments, students can experiment freely without fearing any serious consequences of making mistakes - this property can often make them a preferred choice over real lab sessions. It has also been proved that the use of interactive simulations can support inquiry-based learning [25].

However, previous work has pointed out that many students struggle to learn effectively with interactive simulations [2, 21, 23, 10]. Indeed, navigating through interactive simulations purposefully can be challenging, especially when the number of controls and the level of complexity are high [1]. Therefore, providing adaptive guidance and feedback has the potential to improve students' learning outcomes. However, interactive simulations are often highly complex systems with a practically unlimited number of possible learning paths. Hence, defining (un-)productive inquiry behaviour in such environments is still an open issue.

Prior work has focused on identifying the key factors of productive inquiry behaviour in interactive systems from student log data by leveraging sequence mining and clustering techniques. For instance, [26] applied information theoretic sequence mining to identify behavioural differences of students using a virtual simulation for electronic circuits. In [4], the authors trained binary classifiers and logistic regression models on log data from virtual environments to categorise students by their science inquiry skills. Similarly, [27] used student log data as input for a linear regression model to predict the conceptual understanding students acquired after using physics and chemistry simulations. Latent Class Analyses models were able to identify different profiles in inquiry performance in two PISA science assessments involving interactive simulations [30]. The work in [20] employed clustering techniques to analyse students' inquiry strategies in an interactive simulation on circuit construction. Log data was manually tagged as ground truth to train a classifier on successive inquiry behaviour in [17]. Other research focused on the teacher and provided them with a dashboard displaying the mined student sequences [29]. A rule-based approach for classifying students' problem solving strategies in an interactive Chemistry simulation is described in [14]. In recent work, [32] assessed the effectiveness of data mining techniques guided by knowledge gained from qualitative observations to predict student's problem-solving skills in an interactive simulation.

However, most of the existing approaches have performed *a posteriori* analyses, requiring the complete log data entries as input to the models. This represents a major limitation, since it does not allow the system to provide feedback to the students while they are still interacting with the simulation. Their high complexity and lack of predefined learning trajectories makes building a student model for these environments a challenge. A popular approach to adaptive interventions in open ended learning environments is the use of a clustering-classification framework [19]. Students are clustered offline based on their interaction sequences and the resulting clusters are interpreted. The classification step is then done online: students are assigned to a cluster in real-time, as explored in [13, 11] for an interactive simulation for electrical circuit construction. Recently, [10] presented a novel pipeline for the early prediction of conceptual understanding in an interactive Physics simulation. They were able to robustly predict students' conceptual understanding in an interactive electronics simulation with only initial fractions of the log data. However, the approach was evaluated using only one data set from a single population (Physics undergraduates). To make such approaches more impactful, it becomes crucial to enable their generalisation across simulations and environments.

In this paper, we propose a new generalisable approach for predicting students' conceptual understanding based on sequential log data from interactive simulations. We leverage students' interaction data to extract meaningful features that encode both the state of the simulation and the action of the student. These features are then used as an input to models based on Gated Recurrent Units (GRUs), to predict students' conceptual understanding. In addition to a standard GRU model, we also propose an extension, the Self-Attention-based Gated Recurrent Unit model. We extensively evaluate our approach on two data sets from different contexts: the first data set stems from students of 10 different vocational schools interacting with a chemistry simulation. The second data set consists of physics undergraduate students using a physics simulation. With our experiments, we aim to address three research questions: 1) To what degree is interaction data within interactive simulations predictive for their obtained conceptual understanding? 2) Can our models predict students' conceptual understanding early on? 3) Is our approach transferable to another context involving a different population and simulation?

Our results demonstrate that our proposed models outperform previous approaches in terms of (early) predictive performance on both data sets. We further show that the inclusion of self-attention enables us to draw conclusions about productive inquiry behaviour in both simulations.

2. LEARNING ENVIRONMENTS

The PhET interactive simulations² allow students to explore natural phenomena using different parameter configurations and ideally infer the underlying principles on their own. A large body of research has focused on analysing students' inquiry strategies using these simulations, usually coupled with an explicit task to be solved using the simulation [35]. To evaluate the (early) prediction models presented in this

²<https://phet.colorado.edu/>

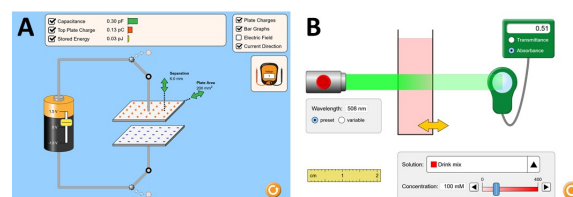


Figure 1: The two learning environments used in this study, the Capacitor Lab (A) and the Beer's Law Lab (B).

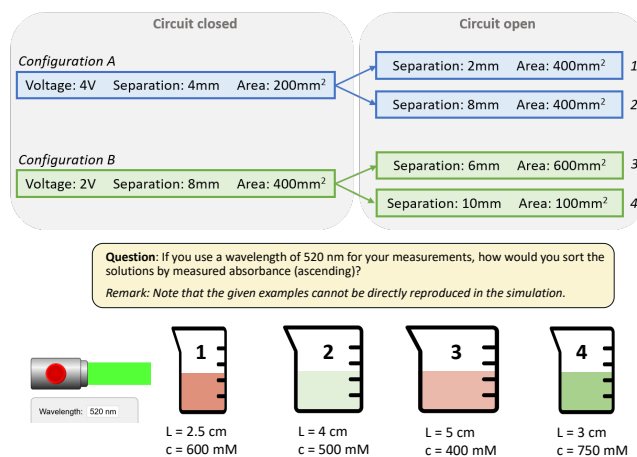


Figure 2: Ranking tasks for the two interactive simulations.

paper, we used a ranking task which we adapted to two different simulation environments. For each ranking task, the values of the variables presented in the task are outside the range provided by the simulation. Hence, students need to inquire the system to uncover the appropriate equations and plug in the numbers as answers in the ranking task.

Capacitor Lab. The PhET Capacitor Lab simulation³ provides students with the possibility to manipulate different parameters of an electric circuit (plate area, plate separation, applied voltage, open/close the circuit), to help them understand how each parameter influences the capacitance of a plate capacitor as well as the energy stored in that capacitor (see Fig. 1A). Previous work [10] designed an inquiry-based learning activity for this simulation using a ranking task. Specifically, students were asked to rank four different configurations according to the stored energy saved in the plate capacitor (see Fig. 2a). The four configurations were created by first loading the plate capacitor using two parameter configurations in a closed circuit (A and B), followed by opening the switch to disconnect the capacitor from the circuit and changing the parameters to obtain the final configurations to be ranked (1, 2, 3, and 4). The authors also suggested a tree-based approach to label the 24 possible solutions based on conceptual understanding. For this paper, we will use the binary labels suggested by [10] to divide students into a group with an **advanced understanding** of stored energy (i.e., for both open and closed circuits) and a second group with a conceptual understanding limited to the closed circuit, denoted as **limited understanding**.

³<https://phet.colorado.edu/en/simulations/capacitor-lab>

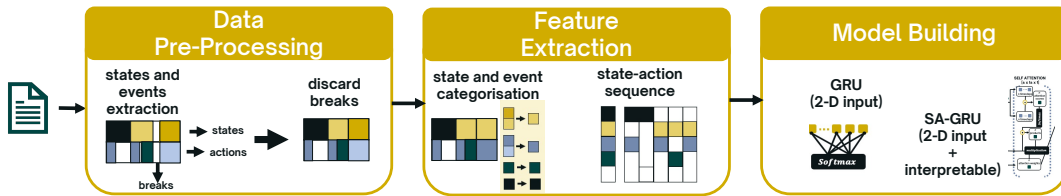


Figure 4: Our model building pipeline consists of three steps: we first extract the logs into sequences of events and states. We then categorise the states and events to build state-action sequences. Finally we feed the features into variations of GRU models.

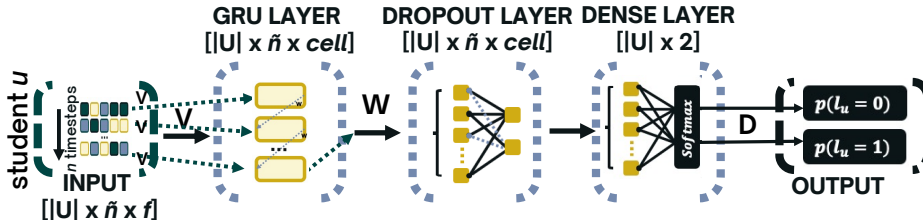


Figure 5: Architecture of our GRU model with one unrolled GRU layer: the input features are fed sequentially into the recurrent cells of the GRU layer. The output shape of each layer is given: $|U|$ denotes the number of users, $\tilde{n} = \max_{u \in \mathcal{U}} \tilde{n}_u$, $cell$ the number of GRU cells, and f the number of features in the input matrix.

can occur naturally throughout a student’s interaction when the mouse moves from one element to the other [31]. The latter might introduce undesired noise and hinder student classification over their conceptual understanding, not unlike stop words negatively influence sentiment analysis tasks in the domain of natural language processing, as an example [15]. To maximise the retention of informative periods of inactivity, we therefore discard for each student $u \in \mathcal{U}$ their 60% of shortest breaks. We are left with $R_u = \{(e_0, s_0, d_0), \dots, (e_{\tilde{n}_u-1}, s_{\tilde{n}_u-1}, d_{\tilde{n}_u-1})\}$ with $\tilde{n}_u \leq n_u$, the final sequence of triplets of events e_m occurring in state s_m and of duration d_m for student $u \in \mathcal{U}$ at time $m \in [0, t_u)$. The new sequence contains \tilde{n}_u event triplets, i.e., *timesteps*.

3.3 Feature Extraction

The data logging of the PhET environments captures any click targeting an “interactive” component as well as the state of the simulation in case one of these “clicks” changes the value of a variable/multiple variables in the system. This fine-grained logging results in high dimensional simulation states s and a high number of different events e . Before creating feature vectors for each time step m , we therefore categorise the obtained events and states into meaningful components. We divide all possible *simulation states* into two groups: a) helpful or b) unhelpful to solve the problem. We then ponder whether some of the *helpful states* can be grouped such that the level of information received by the student is the same independent of the state within the group. We proceed in a similar way for the events. While this categorisation procedure is generalisable over different platforms, whether a simulation state or event is helpful depends on the actual task.

State-Action Categorisation for Capacitor Lab. For the Capacitor Lab activity, we use the state and action categorisation defined in [10]. By applying the above pipeline, we qualify each state $s_m \in (e_m, s_m, d_m)$ into one of the following groups: 1) closed circuit - stored energy displayed, 2) open circuit - stored energy displayed, 3) closed circuit

- stored energy not displayed, and 4) open circuit - stored energy not displayed. Since stored energy is the dependent variable in the ranking task, displaying the stored energy is essential for solving the task. The relationship between the parameters (plate area, plate separation, battery voltage) and stored energy differs for closed and open circuits. Moreover, we map each event $e_m \in (e_m, s_m, d_m)$ into one of the following actions: 1) manipulating battery voltage, 2) manipulating plate area, 3) manipulating plate separation, 4) break, 5) other (any other action).

State-Action Categorisation for Beer’s Law Lab. There are four independent variables that influence the dependent variable *absorbance* of the ranking task: the *laser colour*, the *solution colour*, the *concentration* of the solution, and finally the *width* of the container. Consequently, we categorise each state $s_m \in (e_m, s_m, d_m)$ into one of those 4 groups: 1) green-green (absorbance: displayed, laser colour: green, solution colour: green), 2) green-red (absorbance: displayed, laser colour: green, solution colour: red), 3) absorbance displayed (absorbance: displayed, laser colour: not green and/or solution colour: neither red nor green), 4) not observed (absorbance: not displayed). To solve the ranking task depicted in Fig. 2, students need to understand how the *width* and *concentration* interact with the *absorbance* when the *solution colour* is either red or green and when the *laser colour* is green, i.e., states 1) and 2). The *width* and the *concentration* share a linear dependency with the *absorbance* and are both colour agnostic. As long as the absorbance is observed, those relations can be discovered no matter the colour of the laser or the solution, i.e., state 3. Furthermore, we process each event $e_m \in (e_m, s_m, d_m)$ into one of those categories: 1) width (width slider is moved), 2) concentration (concentration slider is moved), 3) pdf (students are reading the instructions from the ranking task), 4) concentration lab (students are interacting with the second simulation available directly from the Beer’s law lab), 5) breaks, 6) other.

State-Action Sequences. After the categorisation step, each

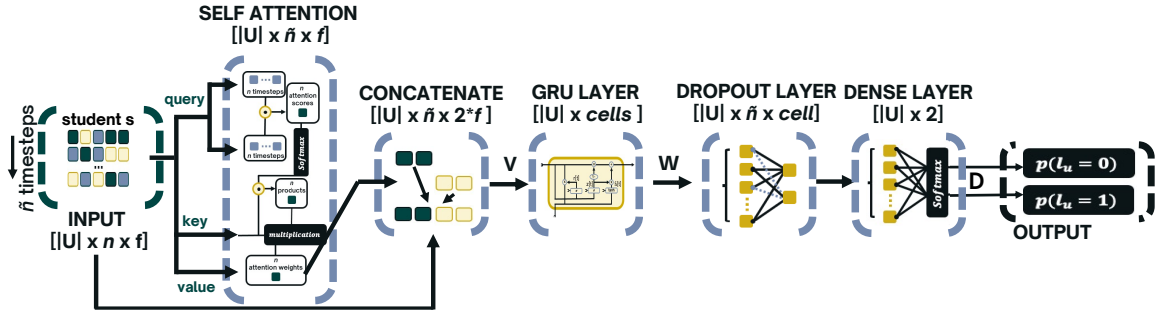


Figure 6: SA-GRU’s architecture with the attention mechanism. The output shape of each layer is given: $|U|$ denotes the number of students, $\tilde{n} = \max_{u \in U} \tilde{n}_u$, f is the number of dimension in the state-action features, and $cells$ is the number of GRU cells.

$e_m \in (e_m, s_m, d_m)$ will have been replaced by e'_m , its corresponding event category, and each $s_m \in (e_m, s_m, d_m)$ will have been replaced by s'_m , its corresponding state mapping. For each timestep m , we encode s'_m as a one-hot-encoded vector v_m^s of c_s cells (with c_s denoting the number of different state categories for the simulation), where all entries are 0 except the one corresponding to s'_m , which is equal to 1. We also transform $e'_m \in (e'_m, s'_m, d_m)$ into a one-hot-encoded vector v_m^e of c_e cells (with c_e denoting the number of different event categories for the simulation), where all entries are 0 except the one corresponding to e'_m , which is equal to d_m . Finally, for each time step m , we replace each triplet (e'_m, s'_m, d_m) by the concatenation of both vectors v_m^s and v_m^e . For each student $u \in U$, the interaction sequence R_u can now be formulated as $\{[v_0^s, v_0^e], \dots, [v_{t_u-1}^s, v_{t_u-1}^e]\}$. We then normalise the sequences on the v_0^e dimension resulting in the final action sequence F_u^{SA} for student $u \in U$.

3.4 Model Building

In this paper, we are interested in creating a conceptual understanding prediction model that can accurately predict the conceptual understanding label L_u for student u , given the extracted features F_u^{SA} . To this end, we rely on two types of models, both based on a *Gated Recurrent Unit* (GRU) network [8]. GRUs are a type of Recurrent Neural Networks (RNNs), whose specificity is to allow the transmission of information over time. To process sequential data, RNNs are composed of r recurrent cells arranged successively such that the output of one becomes the input of the next one, as illustrated on Figure 5. Their relative simplicity enables GRUs to handle smaller datasets (as those in our hands) than the more complex long short-term memory cells (LSTM), which contain an additional output gate [9, 18].

The first type of model is based on a GRU combined with a Dense layer (GRU), as shown in Figure 5. Specifically, the extracted features F^{SA} are fed into a neural architecture composed of a GRU, a dropout layer and a Dense layer (with Softmax activation) having a hidden size of 1. The model outputs the probability the student u will gain advanced (label 1) or limited (label 0) understanding.

The second type of model is based on a Self-Attention-based Gated-Recurrent-Unit (SA-GRU), as shown in Figure 6. In 2014, the concept of an *Additive Attention* layer, also called *Bahdanau attention* was introduced to improve statistical

machine translation [3]. Rather than using it to understand the context of words across languages, we use it to identify the pivotal moments in students’ interactions with the simulation. We first take the raw extracted features F^{SA} as the *query*, *key* and *value* input matrices to our self-attention mechanism implemented as in [16]. We then concatenate the attention output with the input features along each time step and feed this matrix to a GRU layer. Its latest output will directly be given to the dropout layer before passing through the final Dense layer which uses a Softmax activation (analogously to the GRU model). Though more complex than GRU and consequently more subject to overfitting on smaller datasets, SA-GRU’s attention weights can be used to understand what dimension of our features are important for the final prediction, and when that importance is emphasised over time. These aspects are fundamental to enable interpretability on such complex yet effective models.

4. EXPERIMENTAL EVALUATION

We evaluated our (early) prediction models on two data sets collected using the ranking task activities on the PhET Beer’s Law Lab and Capacitor Lab simulations (see Section 2). We used the data set that we collected using the Beer’s Law Lab activity as an *evaluation data set* to assess to what degree student interaction data within the simulation is predictive for their obtained conceptual understanding (RQ1) and whether our models are able to predict students’ conceptual understanding at the end of the task *early* on (RQ2). We then used the data that was collected by previous work [10] using the Capacitor Lab activity as a *validation data set* to assess the transferability of our proposed models to a different population and simulation (RQ3). In the following, we first describe the experimental setup used for our analysis, before describing each experiment in detail.

4.1 Experimental Setup

We compared the predictive performance of our models on the two data sets to baseline models used in prior work [10].

4.1.1 Data Sets

The evaluation and validation data sets were collected in two different classroom experiments (see Table 1).

Evaluation Data Set. Using the Beer’s Law Lab activity (see Section 2), we collected data from 448 laboratory technician apprentices of 10 different vocational schools in a European

	Beer’s Law Lab	Capacitor Lab
Number of students	254	193
Gender	Male: 53% Female: 43% Other: 4%	N/A
Region	A: 70% B: 30%	USA
Education Level	Vocational School	University
Field	Chemistry: 88% Other: 12%	Physics majors: 17% Science/Engineering: 83%
Mean time in simulation	507s	512s
Percentage label 1	44%	51%

Table 1: Statistics of the data sets used for the experiments.

country. The activity was performed directly in the classroom. We recorded students’ interaction logs (clickstream data) as well as their responses to the ranking tasks. For our experiments, we removed participants who did not answer the ranking task (23 students), with an extremely short task duration (less than 2 minutes), or with a low number of events (less than 10 events). The final data set used for the experiment therefore consists of 254 students. Prior to data collection, participants gave their informed consent to sharing the data for research, and all data was recorded in a completely anonymous way. Students could also participate in the learning activity without sharing their data for research. This study was approved by the responsible institutional review board (HREC number: 064-2021).

Validation Data Set. We used the data set collected by [10] on the Capacitor Lab Activity (see Section 2). It contains the interaction logs (clickstream data) and the ranking task responses of 193 undergraduate Physics students of a US university. Students were all in their first year of studies and completed the activity as part of a mandatory homework assignment. Data was recorded in completely anonymous way and students had to give their informed consent prior to data collection. Moreover, students could participate in the learning activity without sharing their data for research. This study was approved by the responsible institutional review board (HREC number: 050-2020).

4.1.2 Baseline Models

We assessed the performance of our sequential models to more simple baseline models. Random forest (RF) models were successfully used in previous work [10] to (early) predict students’ level of conceptual understanding gained through interacting with an interactive simulation. [10] experimented with neural network models and RF models with different type of features and did not find significant differences regarding predictive performance of the different model and feature combinations. We selected RF model with *Action Span* features (RF) as a baseline for our experiments, as this encoding is most similar to our action-state sequences. The *Action Span* features can be easily built from our state and action categories. For each time step m , the encoding of the triplet (e'_m, s'_m, d_m) is similar to a one-hot encoded vector, where all entries are 0 except the one representing $e'_m \times s'_m$ (state-action combination), which is equal to d_m . We therefore obtain a vector v_m of dimensions

$c_s \times c_e$. The raw interaction sequence R_u for student $u \in \mathcal{U}$ then becomes $I_u = \{v_0, \dots, v_{\tilde{n}_u}\}$. We flatten I_u by summing all vectors v_m , resulting in a vector where each cell represents the amount of time in seconds the student spent doing each type of state-action combination. We then normalise the vector by the total duration of the interaction so that the model does not consider the length of each interaction, but rather the proportion of time spent on each condition.

4.1.3 Optimisation Protocol

Both GRU and SA-GRU are relatively sensitive to *seeds* which have an influence on all weight initialisation. To temper the effects of randomness on the models’ training, we used a cross validation with seed mitigation to optimise the neural network models. The usual nested cross-validation optimisation protocol was conducted for the baseline RF model. We evaluated performance of all models using the macro-averaged area under the ROC curve (AUC).

Optimisation for RF. We optimised the RF model using a 10-fold nested cross validation stratified over the labels L_u . We use this step as an evaluation mean for the RF and as a basis for parameter optimisation for the GRU-based models. We ran the inner grid search on the following hyperparameters: the *number of decision trees* [3, 7, 9, 11, 13, 15, 17], the *split criterion* [*gini*, *entropy*], the *maximum depth* [5, 7, 9, 11, 13], the *minimum samples split* [3, 5, 7, 9, 11].

Optimisation for GRU and SA-GRU. For the two sequential models, we performed initial prototyping using a 10-fold nested cross validation (using the same folds as for the RF model) and a small number of seeds. We trained both the GRU and SA-GRU for 150 epochs, using a categorical crossentropy loss and an Adam optimiser. We tuned the following hyperparameters: dropout rate after the GRU layer [0.02, 0.05] and the number of cells [32, 64] in the GRU layers. The initial experiments demonstrated that: the selected hyperparameter values in the inner grid search were the same for a majority of the folds, the categorical cross entropy loss on the inner validation data set tended to converge after 20 – 30 iterations, and predictive performance of the models seemed to vary depending on the seed. We therefore decided to choose a fixed architecture for the GRU and SA-GRU models and limit the number of epochs to 30 for all our experiments. For Beer’s Law Lab the selected parameters were a dropout rate of 0.02 and 32 units for the GRU layer. For the Capacitor Lab, we also use 32 units in the GRU Layer and a dropout rate of 0.05. We evaluated the models using 10-fold cross validation (based on the same outer folds as the RF model). To mitigate the potential effects of randomness, we trained both models over n different seeds and averaged predictive performance across seeds. For our experiments on the full state-action sequences (see Sections 4.2 and 4.4), we used $n = 60$ seeds. Since we observed only minor differences in predictive performance across seeds, we reduced the number of seeds to $n = 10$ for our early prediction experiments (see Sections 4.3 and 4.4).

4.2 RQ1: Full Sequence Prediction

In a first experiment, we investigated whether our proposed models were able to predict students’ level of conceptual understanding based on their interaction data. We used

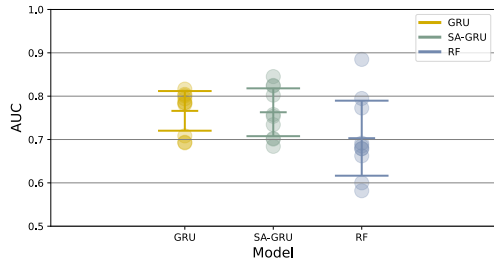


Figure 7: AUC of the GRU, SA-GRU, and RF models on the evaluation data set collected from Beer’s Law Lab. Predictions were made based on the complete sequential interaction data of the students.

the evaluation data set collected with the Beer’s Law Lab simulation to answer RQ1.

Predictive Performance. Figure 7 illustrates the predictive performance in terms of AUC for the GRU, the SA-GRU, and the RF baseline. Both sequential models outperform the RF baseline: using GRU and SA-GRU models we improve the AUC by 10% ($AUC_{GRU} = 0.77$, $AUC_{SA-GRU} = 0.76$, $AUC_{RF} = 0.70$). We also observe that the RF model exhibits a higher variability across folds than the two sequential models. For those, the minimum AUC is close to 0.70 ($AUC_{min,GRU} = 0.69$, $AUC_{min,SA-GRU} = 0.68$). In contrast, performance of the RF model drops below 0.6 based on the fold ($AUC_{min,RF} = 0.58$). We also performed a preliminary analysis on model bias according to protected attributes. Specifically, we grouped the students based on the available protected attributes and computed predictive performance in terms of AUC separately for the different groups. We found that the GRU model is slightly biased towards male students ($AUC_{m,GRU} = 0.78$, $AUC_{f,GRU} = 0.73$), while the two other models do not show any gender bias. All models show a bias in terms of region, with a higher AUC for region A ($AUC_{A,GRU} = 0.78$, $AUC_{A,SA-GRU} = 0.78$, $AUC_{A,RF} = 0.76$) than for region B ($AUC_{B,GRU} = 0.69$, $AUC_{B,SA-GRU} = 0.66$, $AUC_{B,RF} = 0.63$).

Attention Interpretation. While adding self-attention to the GRU does not increase predictive performance of the model (there is no significant difference in the performance of the GRU and SA-GRU model), the self-attention scores enable interpretation of the such black-box models. The heatmap in Fig. 8 shows the normalised attention scores for the SA-GRU model over 150 time steps. For each student $u \in \mathcal{U}$, we extracted the output (dimension: $n_u \times f$) of the self-attention layer of the SA-GRU model (see Fig. 6) and normalised it to make the attention scores comparable between models. We then averaged across all students. The number of students decreases rapidly with an increasing number of time steps (Fig. 7 (top)). In fact, for 90% of the students, $n_u \leq 150$.

We observe that the model seems to pay most attention to two of the states (*green-red* and *no absorbance*). The former state, *green-red*, indicates a green laser colour and a red solution colour while the absorbance is measured. As expected, exploring with this parameter setting is important for solving the ranking task. The latter state, *no absorbance*,

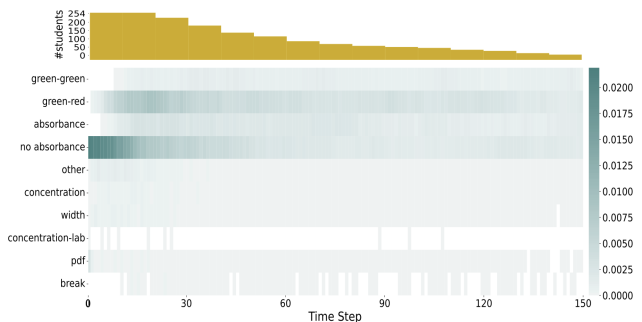


Figure 8: Normalized attention scores for the SA-GRU over 150 time steps (bottom). Number of students with $n_u \geq t$ for $t = 1, \dots, 150$ (top).

indicates that the absorbance (outcome variable of the task) was not displayed, which makes solving the task impossible. Also the two other states (*green-green* and *absorbance*) seem to have phases of higher importance over time. For the *no absorbance* state, we can see that the the model seems to pay less attention to it with an increasing number of time steps. Generally, the actual state of the simulation seems to be considered as more important than a specific action of the student. Regarding the action, the model seems to pay attention to state *pdf*, which refers to students checking the task description and hence the configurations to be ranked. Also experimenting with *concentration* and *width* seems to have some importance. Interestingly, the importance of taking a *break* varies over time steps. Finally, the model does not pay attention to the *concentration-lab* action.

In summary, this first experiment demonstrated that our proposed models yielded a 10% increase in AUC over the RF approach suggested in previous work [10]. Furthermore, the sequential models also exhibit a lower variance across folds. Finally, by adding self-attention to a GRU model, we were able to provide interpretations of what type of student exploration behavior will lead to successfully solving the task.

4.3 RQ2: Early Prediction

In our second experiments, we were interested in predicting students’ level of understanding *during* their interaction with the simulation. We therefore did not use all students’ full interaction sequences, but used partial interactions up to a time step l when the student went over that limit. Therefore to build the features $F_u^{l,SA}$ for student $u \in \mathcal{U}$, we only used the interactions I_u^l as a basis for building the features (see also Section 3.3). We trained all our models for $l = 30, 40, 50, 60$. We did not make predictions earlier as we did not expect a model to be able to accurately predict conceptual understanding after only a few student actions. We further limited $l \leq 60$ as the number of students decreases for higher time steps (see also Fig. 8 (top) which indicates the amount of remaining students per time step). At each time step l , we made predictions for students with state-actions sequences at each length l (e.g. $\tilde{n}_u \geq l$). For students with shorter sequences (e.g., $\tilde{n}_u < l$), we used the last available prediction. We used the evaluation data set collected with Beer’s Law Lab to answer RQ2.

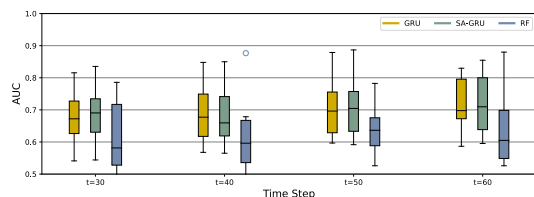


Figure 9: Early predictive performance in terms of AUC for RF, GRU, and SA-GRU for the evaluation data set collected using Beer’s Law Lab for $t = 30, 40, \dots, 60$.

Predictive Performance. Fig. 9 illustrates the AUC of the different models for early prediction at $t = 30, 40, 50, 60$. We again observe that the RF baseline model performs inferior to the sequential models. It achieves an AUC of 0.59 after 30 time steps and manages to only slightly improve over time with an AUC of 0.61 after 60 time steps. The AUC of the RF is 0.70 for full sequence prediction (see Section 4.2), we therefore observe a strong increase in predictive performance when observing full sequences. It thus seems that the RF classifiers is not able to extract strong signal from early observations. The use of more elaborate sequential models leads to an up to 15% increase in AUC ($AUC_{30,GRU} = 0.67$, $AUC_{30,SA-GRU} = 0.68$, $AUC_{60,GRU} = 0.72$, $AUC_{60,SA-GRU} = 0.72$). Similar to the full sequences case, RF exhibits a much higher variance across folds than that our proposed models.

Attention Interpretation. Next, we investigated whether attention scores differed between an early prediction and a full sequence prediction model, i.e. whether the model paid attention to different features when making early predictions.

Figure 10 shows the normalised attention scores for the SA-GRU model trained to predict at time step $t = 30$. The scores in the heatmap were calculated following the same procedure as for the full sequences case (see Section 4.2). Similar to our findings on full sequences, we observe that the states generally seem to be assigned higher scores than the actions. While we again observe that the *no absorbance* and *green-red* states have relatively higher scores, we also observe that these scores vary over time. The scores of the *no absorbance* state clearly decrease over time. This finding might suggest that many students realise during their first 10 – 15 interactions with the simulation that the absorbance display should be turned on (by default, transmittance is displayed). In contrast, the scores of the *green-red* state increase over time, indicating that after being able to observe the outcome variable (absorbance), it is important to select the optimal colours for laser (green) and solution (red). For the actions, we observe the same picture as for the full predictions. The *concentration-lab* action gets little attention (it is probably also a rare action), while manipulating *width* and *concentration* gets similar attention over time. Checking the task description is mainly important early on. The model does not pay attention to *break* for the first 10 time steps. We hypothesise that “thinking” breaks do not occur often during initial interaction when students are still focused on understanding the task and have not yet started their exploration.

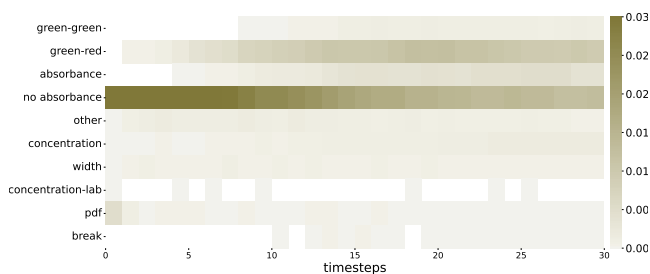


Figure 10: Normalised attention scores for SA-GRU trained for predicting at $t = 30$ time steps for Beer’s Law Lab.

In summary, our second set of experiments showed that in case of early prediction, the use of a GRU or SA-GRU model leads to even larger performance increases than in case of full sequences. Depending on the time step, we observe improvements in AUC of 15% compared to the RF baseline. Furthermore, the RF again shows high variability with drops in AUC down to 0.50. We are again able to provide meaningful interpretations of the attention weights, which is a promising basis for targeted intervention.

4.4 RQ3: Generalisation

In our last experiment, we were interested in assessing the generalisability of the developed models. While both learning activities were based on a ranking task, the underlying simulation as well as the population are different. We therefore evaluated our models on the transfer data set collected from the Capacitor Lab simulation.

Prediction on Full Sequences. We first evaluated the generalisability of the different model by predicting on the full interaction sequences of the validation data set. Figure 11 illustrates the predictive performance in terms of AUC for the GRU, the SA-GRU, and the RF baseline. We observe that all the models achieve a very high AUC for the Capacitor Lab data set. The sequential models achieve an AUC of 0.96 (GRU) and 0.96 (SA-GRU) respectively, while the AUC of the RF baseline is 0.95. Again the RF baseline model shows higher variability, achieving a minimum AUC of 0.86. For the GRU and SA-GRU models, we achieved a minimum AUC of 0.9 and 0.89 respectively. For Capacitor Lab, we ran the same bias analysis on protected attributes as for Beer’s Law Lab, showing that models are not biased with respect to field (the only known protected attribute for this data set).

While the SA-GRU model does not outperform the other two models, we again extracted the attention weights of this model and interpreted them. Figure 12 shows the normalised attention scores of the SA-GRU models over a 100 time steps. We chose to limit to 100 time steps as for 90% of the students $\tilde{n}_u \leq 100$. Again, the scores displayed in the heatmap were calculated using the procedure described in Section 4.2. Similar to our findings on Beer’s Law Lab, we observe that the states (*stored energy*, *closed circuit*) receive higher attention than the actions. For *stored energy*, attention scores increase up to time step 30 and then decrease again after time step 60. We assume that students needed some time to explore the simulation and figure out that the stored energy

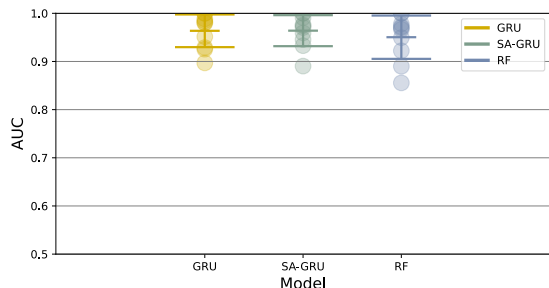


Figure 11: AUC of the GRU, SA-GRU, and RF models on the validation data set collected from CapacitorLab. Predictions were made based on the full interaction data of the students.

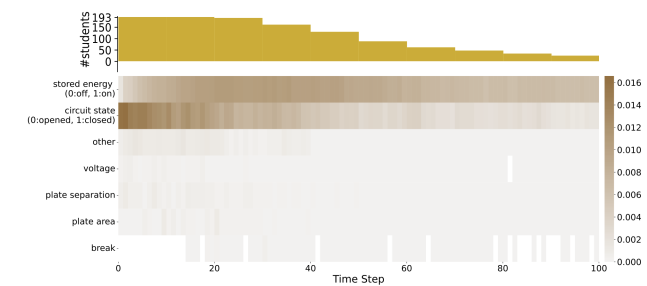


Figure 12: Normalised attention scores for CapacitorLab for the SA-GRU over 150 time steps (bottom). Number of students with $n_u \geq t$ for $t = 1, \dots, 150$ (top).

display needs to be turned on (it is by default turned off in the simulation). For the *closed circuit* state, attention scores decrease over time, but there are always phases of increased activity. In order to be able to solve the ranking task, it is important to experiment with the closed as well as the open circuit, as detected by the model. The model also seems to pay attention to actions related to *voltage*, *plate area*, and *plate separation*, which are the other three components in the simulation that influence the stored energy. Moreover, *breaks* are important for different phases within the sequences. Hence, the model seems to detect when students take a thinking break and considers that as important.

Early Prediction. Next, we also trained the different models for early prediction on Capacitor Lab. Figure 13 shows the AUC of the evaluated models for prediction on $t = 30, 40, 50, 60$ time steps. We see that the AUC of the two sequential model is close to 0.80 already after observing 30 student interactions ($AUC_{30,GRU} = 0.8$, $AUC_{30,SA-GRU} = 0.81$). Predictive performance of the RF model is slightly lower ($AUC_{30,RF} = 0.76$). After 60 time steps, all the models achieve an AUC larger than 0.8 ($AUC_{60,GRU} = 0.9$, $AUC_{60,SA-GRU} = 0.9$, $AUC_{60,RF} = 0.88$). We observe that all three models show a similar variability under this task.

In summary, our results demonstrate that the proposed models can be transferred to a different learning environment and population. In case of full interaction data, the performance of the models is inline with the RF baseline, in case of early prediction, the sequential models outperform the base-

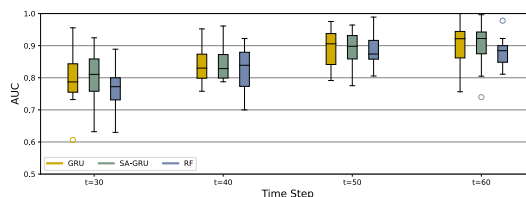


Figure 13: AUC of the GRU, SA-GRU, and RF models for early prediction ($t = 30, 40, \dots, 60$) on Capacitor Lab.

line. Furthermore, the addition of self-attention again enabled us to identify the key points in the exploration process.

5. DISCUSSION AND CONCLUSION

Interactive simulations used for inquiry-based learning activities are increasingly gaining traction as part of science education at all levels. It has thus become essential to understand how students navigate in such environments in order to provide them with targeted and well-timed feedback. To address this issue, we introduced a new data-driven approach to predict students’ conceptual understanding after solving inquiry-based learning tasks with interactive simulations. We leveraged log data directly recorded in these environments to extract meaningful features that were then used as input to models based on Gated Recurrent Units (GRU) to predict their conceptual understanding. In addition to a standard implementation of a GRU model, we also explored an extension, the Self-Attention-based Gated Recurrent Unit (SA-GRU) model, and compared their performance against a random forest (RF) classifier presented in prior work [10]. We evaluated our approach on two data sets collected from different populations (vocational and undergraduate students) who used two different environments (a chemistry and a physics simulation). We aimed at answering to three research questions: 1) To what degree is student interaction data within interactive simulations predictive for their obtained conceptual understanding? 2) Can our models predict students’ conceptual understanding early on? 3) Is our approach transferable to another context involving a different population and simulation?

To answer the first research question, we trained the proposed models using data collected from 254 first, second and third year lab technician apprentices who solved a ranking task using the PhET Beer’s Law Lab simulation. Using the complete log sequences of the students, both the GRU and SA-GRU models yielded AUC values of 0.77 and 0.76 respectively, outperforming the RF baseline model presented in previous work [10] by 10%. For both models, we also observed lower variance across training folds, indicating a higher robustness compared to the RF baseline. Previous research has highlighted that along with predictive accuracy, it is the interpretability of the models that can make a significant difference in improving student learning outcomes [12]. While adding the self-attention layer to the GRU models did not further improve their predictive performance, our analyses illustrated that they significantly enhanced the interpretability of the model results. For instance, analysing the attention scores of the SA-GRU model, we observed that the state s in which an action a is conducted is more important

than the nature of action a itself to predict conceptual understanding. Experimenting in a setting where the variable of interest is observed, and where the environment settings correspond to those present in the problem, is influential.

Our findings are in line with those of [32], who suggested that knowledge-grounded features that integrate qualitative observations are better predictors than purely actions-based features. From the attention score analysis, we also observed that taking breaks showed an interesting pattern with an alternating importance for the model over time. This result seems to indicate that taking breaks at specific moments during the activity may be predictive of the acquired understanding. These results are very much in line with those reported in previous work [26, 7], who illustrated that recurrently taking breaks for reflection and planning can be a powerful inquiry strategy in interactive simulations.

To address the second research question, we then trained the models with only initial sequences of the students' log data in order to evaluate their capabilities to early predict students' conceptual understanding. Naturally, the performance of all models dropped compared to the training on the full sequences. However, both the GRU and SA-GRU outperformed the RF baseline also in early prediction, with AUC values that were up to 15% higher and with lower variance across training folds. After only 30 time steps in the simulation, AUC values for the GRU and SA-GRU models already reached 0.67 and 0.68 respectively. Moreover, the analysis of the attention scores of the SA-GRU model showed that already after 30 time steps, meaningful interpretations could be made similar to those in the case of the models trained on the full sequences. This is a crucial result, since it enables the implementation of targeted interventions that could, for instance, support struggling students early on while they are still solving the tasks. Such interventions could potentially be implemented as automatic feedback that is directly displayed in the simulation, similar to approaches that have been adopted previously for other computer-based learning environments [5]. Another possibility could be to display the insights gained from the models to teachers using real-time dashboards [29, 22], to support them in preparing interventions for individual students and/or the entire class.

Finally, to answer our third research question we evaluated our approach on a second data set from [10]. It comprised log data from a different simulation (the PhET Capacitor Lab) that was used by a different population (undergraduate students attending a physics class). The goal of this analysis was to demonstrate that the functioning of our models is not restricted to a single context, but could be transferred to other populations and environments. In the case of training on the full sequences of log data, all three models performed very well with AUC values above 0.95. Since in this case the RF baseline already showed a good performance, the use of the GRU and SA-GRU models only increased the AUC by 1%. Similar to the results from the Beer's Law Lab data set, we found higher importance for states (stored energy toggled and circuit closed) in the attention scores of the SA-GRU model. The high attention scores for the stored energy state could be explained by the observations made in previous work using the same simulation. [22] found that despite the stored energy being the target outcome variable

of their task, not all students toggled the function to display it. Similarly, the stored energy state could have served as a useful feature to detect unproductive behaviour in our task. Moreover, while the design of the ranking tasks for both simulations was similar, the use of different circuit states (closed or open) in the Capacitor Lab might be more discriminatory for student performance than the color states in Beer's Law Lab. This could explain why for the Capacitor Lab the RF baseline also performed very well. Finally, when looking at early prediction, we observed again that GRU and SA-GRU exhibited better performances compared to RF despite the differences being smaller than for Beer's Law Lab. Variances across folds, however, were similar across models.

Our findings represent an important step towards a better understanding and modelling student behaviour in ranking-based inquiry learning activities within interactive simulations. Though the presented task is very specific, its key characteristics can be retained and transferred onto other interactive environments. Furthermore, the capabilities of the presented models to detect (un-)productive behaviour early on pave the way for more targeted interventions carried out by teachers or the system itself.

Nevertheless, this study also comes with certain limitations. First, the chosen labels for the Beer's Law Lab are only able to separate *struggling students* from *more advanced students*, giving us an indication of whether students can conduct inquiry rather than whether they have understood Beer's Law. Then, our preliminary bias analysis has indicated that some models show a slight gender bias (in case of the Beer's Law Lab simulation) and all models exhibit a geographic bias (also for the Beer's Law lab simulation). Therefore, further work on bias mitigation is needed. This includes understanding where the differences in predictions come from, as well as identifying any other confounding factors. Furthermore, while the presented methods were applied to two different data sets involving different contexts to illustrate its transferability, it can not be guaranteed that they are unreservedly applicable to any other inquiry-based learning situation involving interactive simulations. More research including larger and more diverse samples as well as different educational levels, simulations, and learning tasks are needed to consolidate the generalisability of our methods. This is particularly important in order to mitigate the risks of algorithmic bias that could be introduced. Furthermore, it needs to be acknowledged that measuring students' conceptual knowledge by means of a single outcome variable (i.e., the final ranking submitted) might be too limiting. While the number of possible rankings could be considered sufficiently high to filter out the effect of random answers, other measuring tools (such as quizzes or interviews) could be included to complement the analyses. Finally, it should be emphasised that before any interventions based on the proposed methods are being introduced, it is imperative that the main stakeholders (i.e., teachers and students) are involved. Only in this way the potential educational impacts of such approaches can be assessed, implying both positive and negative consequences for the concerned groups.

6. REFERENCES

- [1] W. K. Adams, A. Paulson, and C. E. Wieman. What levels of guidance promote engaged exploration with interactive simulations? In *AIP*, volume 1064-1, pages 59–62. American Institute of Physics, 2008.
- [2] L. Alfieri, P. J. Brooks, N. J. Aldrich, and H. R. Tenenbaum. Does discovery-based instruction enhance learning? *Journal of educational psychology*, 103(1):1, 2011.
- [3] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.
- [4] R. S. Baker, J. Clarke-Midura, and J. Ocumpaugh. Towards general models of effective science inquiry in virtual performance assessments. *Journal of Computer Assisted Learning*, 32(3):267–280, 2016.
- [5] A. T. Bimba, N. Idris, A. Al-Hunaiyyan, R. B. Mahmud, and N. L. B. M. Shuib. Adaptive feedback in computer-based learning environments: a review. *Adaptive Behavior*, 25(5):217–234, 2017.
- [6] W. V. Bo, G. W. Fulmer, C. K.-E. Lee, and V. D.-T. Chen. How do secondary science teachers perceive the use of interactive simulations? the affordance in singapore context. *Journal of Science Education and Technology*, 27(6):550–565, 2018.
- [7] E. Bumbacher, S. Salehi, M. Wierzchula, and P. Blikstein. Learning environments and inquiry behaviors in science inquiry learning: How their interplay affects the development of conceptual understanding in physics. *EDM*, 2015.
- [8] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.
- [9] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.
- [10] J. M. Cock, M. Marras, C. Giang, and T. Kaser. Early prediction of conceptual understanding in interactive simulations. In *EDM*, pages 161–171, 2021.
- [11] C. Conati, L. Fratamico, S. Kardan, and I. Roll. Comparing representations for learner models in interactive simulations. In *AIED*, pages 74–83, 2015.
- [12] C. Conati, K. Porayska-Pomsta, and M. Mavrikis. Ai in education needs interpretable machine learning: Lessons from open learner modelling. *arXiv preprint arXiv:1807.00154*, 2018.
- [13] L. Fratamico, C. Conati, S. Kardan, and I. Roll. Applying a framework for student modeling in exploratory learning environments: Comparing data representation granularity to handle environment complexity. *JAEID*, 27(2):320–352, 2017.
- [14] Y. K. Gal, O. Uzan, R. Belford, M. Karabinos, and D. Yaron. Making sense of students’ actions in an open-ended virtual laboratory environment. *Journal of Chemical Education*, 92(4):610–616, 2015.
- [15] K. V. Ghag and K. Shah. Comparative analysis of effect of stopwords removal on sentiment classification. In *International Conference on Computer, Communication and Control*, pages 1–6. IEEE, 2015.
- [16] D. Ghosal, M. S. Akhtar, D. Chauhan, S. Poria, A. Ekbal, and P. Bhattacharyya. Contextual inter-modal attention for multi-modal sentiment analysis. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3454–3466, 2018.
- [17] J. Gobert, M. Sao Pedro, J. Raziuddin, and R. S. Baker. From Log Files to Assessment Metrics: Measuring Students’ Science Inquiry Skills Using Educational Data Mining. *Journal of the Learning Sciences*, 22(4):521–563, 2013.
- [18] N. Gruber and A. Jockisch. Are gru cells more specific and lstm cells more sensitive in motive classification of text? *Frontiers in artificial intelligence*, 3:40, 2020.
- [19] S. Kardan and C. Conati. A framework for capturing distinguishing user interaction behaviors in novel interfaces. In *EDM*, pages 159–168, 2011.
- [20] S. Kardan, I. Roll, and C. Conati. The Usefulness of Log Based Clustering in a Complex Simulation Environment. In S. Trausan-Matu, K. E. Boyer, M. Crosby, and K. Panourgia, editors, *Intelligent Tutoring Systems*, pages 168–177, 2014.
- [21] P. Kirschner, J. Sweller, and R. E. Clark. Why unguided learning does not work: An analysis of the failure of discovery learning, problem-based learning, experiential learning and inquiry-based learning. *Educational Psychologist*, 41(2):75–86, 2006.
- [22] D. López-Tavares, K. Perkins, S. Reid, M. Kauzmann, and C. Aguirre-Vélez. Dashboard to evaluate student engagement with interactive simulations. In *Physics Education Research Conference Proceedings*, 2018.
- [23] R. E. Mayer. Should there be a three-strikes rule against pure discovery learning? *American psychologist*, 59(1):14, 2004.
- [24] K. W. McElhaney and M. C. Linn. Investigations of a complex, realistic task: Intentional, unsystematic, and exhaustive experimenters. *Journal of Research in Science Teaching*, 48(7):745–770, 2011.
- [25] E. B. Moore, T. A. Herzog, and K. K. Perkins. Interactive simulations as implicit support for guided-inquiry. *Chemistry Education Research and Practice*, 14(3):257–268, 2013.
- [26] S. Perez, J. Massey-Allard, D. Butler, J. Ives, D. Bonn, N. Yee, and I. Roll. Identifying productive inquiry in virtual labs using sequence mining. In *AIED*, pages 287–298. Springer, 2017.
- [27] S. Perez, J. Massey-Allard, J. Ives, D. Butler, D. Bonn, J. Bale, and I. Roll. Control of variables strategy across phases of inquiry in virtual labs. *JAIED*, pages 271–275, 2018.
- [28] N. Rutten, W. R. Van Joolingen, and J. T. Van Der Veen. The learning effects of computer simulations in science education. *Computers & education*, 58(1):136–153, 2012.
- [29] D. L. Tavares, K. Perkins, M. Kauzmann, and C. Aguirre Velez. Towards a teacher dashboard design for interactive simulations. In *Journal of Physics: Conference Series*, volume 1287-1, page 012055. IOP Publishing, 2019.
- [30] N. Teig, R. Scherer, and M. Kjærnsli. Identifying patterns of students’ performance on simulated inquiry tasks using pisa 2015 log-file data. *Journal of Research in Science Teaching*, 57(9):1400–1429, 2020.

- [31] K. Wang, K. Nair, and C. Wieman. Examining the links between log data and reflective problem-solving practices in an interactive task. In *LAK*, pages 525–532, 2021.
- [32] K. D. Wang, S. Salehi, M. Arseneault, K. Nair, and C. Wieman. Automating the assessment of problem-solving practices using log data and data mining techniques. In *L@S*, pages 69–76, 2021.
- [33] C. E. Wieman, W. K. Adams, and K. K. Perkins. Phet: Simulations that enhance learning. *Science*, 322(5902):682–683, 2008.
- [34] H.-K. Wu and Y.-L. Huang. Ninth-grade student engagement in teacher-centered and student-centered technology-enhanced learning environments. *Science education*, 91(5):727–749, 2007.
- [35] F. Yehya, A. Barbar, and S. Abou-Rjelil. Learning with simulations: Influence of a computer simulation with hand-on activities on students' learning of the physics capacitors' concepts. *Research in Social Sciences and Technology*, 4(1):1–29, 2019.

Toward Better Grade Prediction via A2GP - An Academic Achievement Inspired Predictive Model

Wei Qiu
Nanyang Technological
University
qiuwei@ntu.edu.sg

S. Supraja
Nanyang Technological
University
supraja.s@ntu.edu.sg

Andy W. H. Khong
Nanyang Technological
University
andykhong@ntu.edu.sg

ABSTRACT

Predicting student performance in an academic institution is important for detecting at-risk students and administering early-intervention strategies. We propose a new grade prediction model that considers three factors: temporal dynamics of prior courses across previous semesters, short-term performance consistency, and relative performance against peers. The proposed architecture comprises modules that incorporate the attention mechanism, a new short-term gated long short-term memory network, and a graph convolutional network to address limitations of existing works that fail to consider the above factors jointly. A weighted fusion layer is used to fuse learned representations of the above three modules—course importance, performance consistency, and relative performance. The aggregated representations are then used for grade prediction which, in turn, is used to classify at-risk students. Experiment results using three datasets obtained from over twenty thousand students across seventeen undergraduate courses show that the proposed model achieves low prediction errors and high F1 scores compared to existing models that predict grades and thereafter identifies at-risk students via a pre-defined threshold.

Keywords

Grade prediction, machine learning, attention mechanism, long short-term memory network, graph convolutional network

1. INTRODUCTION

Learning analytics involves the process of collecting, analyzing, and reporting of data generated by learners in an education setting. It optimizes learning and the environment by gaining insights into the learning behavior and/or learner achievements [39]. Among the several sub-disciplines that learning analytics transcends across, prediction of academic performance has received increasing attention in recent years and remains one of the most challenging tasks. Grade prediction plays a central role in the development

of data-informed approaches for early-intervention strategies and it is therefore important to achieve a low prediction error—errors leading to high false alarms will result in reduced morale and inefficient allocation of resources while missed detection often results in sustained poor performance [29]. After grade prediction, at-risk students are identified as those whose performance satisfy a pre-defined set of conditions (e.g., those who score below the passing mark in one or more courses).

1.1 Existing Works for Grade Prediction

Prediction of academic performance in the form of grade point averages [15], examination grades [13], or academic achievements [25] can be achieved via a variety of sources. These sources include (but not limited to) online learning activities [22, 28, 44], co-curricular activity records [8], demographics [38], and course grades obtained from previous semesters [13, 23, 32, 34]. While online learning offers numerous opportunities for the exploitation of data associated with learning behaviors (in the form of clickstreams and/or online assessment results) [46], many academic institutions still rely on face-to-face instructions for some courses. Extraction of learning behaviors for these courses via audio/visual capturing devices may present challenges in terms of technological capability and privacy concerns. In addition, co-curricular activity records and demographic profiles may not be readily available due to the general personal data protection policies [3]. Therefore, grade prediction using past examination records as useful features [45] has been the main focus in recent years since examination results are often made accessible to policy makers, administrators, instructors, and student care support personnel involved in developing and administering intervention strategies.

Machine learning techniques have been proposed to predict grades of a given (pilot) course based on those achieved in historical (prior) courses. These models exploit the temporal dynamics of student performance across semesters from two aspects—consistency in academic performance and course importance [47]. These aspects have been modeled using a sequential model and the attention mechanism [24], respectively. Sequential models such as the long short-term memory (LSTM) has been applied to model long-term dependencies in online interaction [17] and to predict the grade point average of a given semester from marks obtained across various courses [33]. More recently, as opposed to predicting the aggregated performance for a semester, the LSTM was trained to predict the grade of each course [13]. In

W. Qiu, S. Supraja, and A. W. H. Khong. Toward better grade prediction via A2GP - an academic achievement inspired predictive model. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 195–205, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852984>

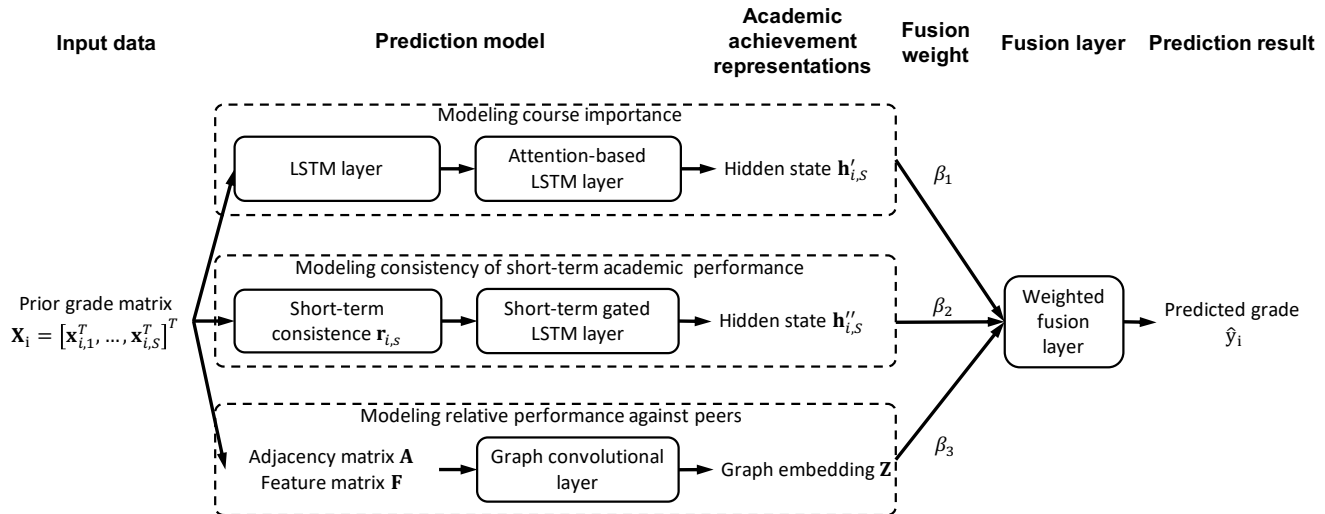


Figure 1: The proposed academic achievement-based grade prediction (A2GP) architecture.

this model, a vector representation of course grades achieved in each of the previous semester was used as input for the LSTM model.

Notwithstanding the above, a course-specific regression model that predicts the grade of a pilot course as a sparse linear combination of prior grades has been proposed [35]. Results presented highlight the detrimental effects of missing regressors for students who have yet attempted an “important” prior course. While modeling of the temporal dynamics of student performance along with the incorporation of the attention mechanism for grade comparison between students has been proposed [30], the intrinsic formulation of LSTM limits its ability to model short-term consistency [21]. Despite the use of knowledge distillation [27], such models do not consider peer performance among students.

In recent years, graph convolutional network (GCN) have been employed to generate meaningful feature representations. In contrast to the use of grade vectors in temporal modeling approaches, these representations model the transitions of grade distributions between courses across semesters [12]. Here, the performance of each student is considered for courses taken consecutively. More recently, nodes representing either students or courses have been used to construct student-course, student-student, and course-course graphs [23]. These graphs consist of edge links computed via grade distribution similarities; they do not model both the long- and short-term sequential information of each student.

While the above techniques achieve good prediction performance, the models are optimized independently and do not consider all the above-mentioned aspects jointly. A holistic approach toward predicting academic performance is important and motivated, in part, by Walberg’s theory of educational productivity. Apart from external variables such as quality of instruction and climate, student-centric variables that include prior achievement and student cognitive capacity will influence the academic performance of an individ-

ual [43].

1.2 The Proposed Model Architecture

Inspired by student-centric factors highlighted in Section 1.1, we propose an academic achievement-based grade prediction (A2GP) architecture that jointly models the (i) importance of prior courses, (ii) short-term consistency in academic performance across previous semesters, and (iii) benchmarking of student performance relative to their peers. With reference to Figure 1, the first module of the proposed architecture comprises an attention-based LSTM network that encodes the influence of prior course grades on the pilot course. This module is based on existing sequential models (such as LSTM) that have been employed to capture the temporal dynamics of past academic performance [13]. These models are motivated by studies that have established the association between course orderings and academic performance [9, 26]. Such an association is not surprising given that the constructivist approach has often been adopted for curriculum design, resulting in the influence of various prior courses on a pilot course [37]. Such a constructivist strategy has also shown to be effective in terms of academic achievement [2, 18] and improving content mastery that requires higher cognitive levels [1]. Temporal modeling of performance using LSTM is also in line with the Tinto’s Student Integration Model which posits that persistence in higher education is a temporal process [5], i.e., the ability to achieve learning outcomes of fundamental courses will influence that of other advanced or related courses. This is further justified if the prior course serves as a pre-requisite for the pilot course.

Compared to existing models that model long-term dynamics of academic achievements, the second module consists of a new short-term gated LSTM (STG-LSTM) that models short-term consistency in academic performance for each student. This module is motivated by the need to consider academic momentum that highlights the influence of workload (which varies across courses and semesters) on academic performance and the achievement of learning out-

comes [14]. Short-term consistency may also arise from academic performance being highly dependent on multiple (yet often convoluted) factors such as socio-economic, psychological, and environmental conditions that a student may face in the recent (past) semesters [36]. From developmental perspective, knowledge inquiry is known to evolve over a series of micro-development resulting in short-term variations in performance [7]. This also aligns with findings that demonstrate the positive effect of mastery and performance goals on short-term and long-term consequences of student achievement [10]. The identification of these patterns would therefore lead to more effective grade prediction.

Beyond representing the performance of a student over time, students are often deemed as at-risk if their performance is consistently below par compared to their peers. In particular, for courses perceived as easy which most students achieve a high grade, achieving a reasonable grade (e.g., Grade C) may still constitute as at-risk when most peers achieved a Grade A. Conversely, a Grade C may not be inferred as being at-risk when most peers achieved similar (or lower) grades for a course perceived by most as challenging. Accounting for such benchmarking of grades is important since such relative performance has shown to achieve lower grade prediction bias than one based on absolute grades [4, 41, 42]. In light of the above, the third module involves a graph convolutional network that models grade differences between student pairs across all prior courses taken by them. This module exploits information derived from students who perform similarly/dissimilarly across commonly taken courses and models such representation that describes the relative performance between students.

For grade prediction, learned academic achievement representations associated with temporal dynamics of past performance, short-term performance consistency of an individual, and relative performance against peers are synthesized via a weighted fusion layer. We formulate a learnable parameter in this layer that determines the relative emphasis of factors influencing the pilot course grade. Learning the weightings for these academic achievement representations is important to model the underlying characteristics of a dataset that contribute to the joint optimization of the model. Performance of the proposed A2GP architecture is evaluated over three student performance datasets obtained over seventeen courses from over twenty thousand students in a university. Results obtained highlighted that the A2GP model improves the performance of LSTM and GCN by 19.0% and 63.3%, respectively, in terms of F1 score for at-risk classification.

This paper is organized as follows: the problem statement and background formulations are described in Section 2. Technicalities of the proposed A2GP model are detailed in Section 3. Details of the datasets, as well as, the comparison analysis with discussions are described in Section 4 while Section 5 concludes the paper.

2. PRELIMINARIES

2.1 Problem Statement

The problem of grade prediction and at-risk detection from prior grades can be described by defining, for each student index i , the exam grade $x_{i,l,s}$ achieved for a prior course l

during semester s . Denoting L as the total number of prior courses, a $1 \times L$ grade vector for semester s is given by

$$\mathbf{x}_{i,s} = [x_{i,1,s}, \dots, x_{i,L,s}], \quad (1)$$

where $x_{i,l,s} = \phi$ is a null element corresponding to an unregistered course l in that semester. The prior course grades of a student across S number of semesters under consideration can then be represented as an $L \times S$ matrix

$$\mathbf{X}_i = [\mathbf{x}_{i,1}^T, \dots, \mathbf{x}_{i,S}^T]. \quad (2)$$

With the above, columns of \mathbf{X}_i form a sequence of vectors that encapsulates the ability of a student to achieve learning outcomes (measured by grades). Given the database of (student, course, grade) up to semester S , the aim is to predict grades for each student on courses he/she will be enrolling in the coming semester.

2.2 Modeling Long-term Dynamics of Academic Performance using LSTM

To model the academic performance across semesters, \mathbf{X}_i serves as features for the prediction of course grades in the forthcoming examinations [13]. The use of \mathbf{X}_i , therefore, allows the model to account for courses that a student has re-attempted. An LSTM unit in semester s is described by

$$\mathbf{h}_{i,s} = \text{LSTM}(\mathbf{x}_{i,s}, \mathbf{h}_{i,s-1}), \quad (3)$$

where the above compact form is defined by

$$\mathbf{f}_{i,s} = \sigma(W_f \cdot \mathbf{x}_{i,s} + V_f \cdot \mathbf{h}_{i,s-1} + \mathbf{b}_f), \quad (4a)$$

$$\mathbf{u}_{i,s} = \sigma(W_u \cdot \mathbf{x}_{i,s} + V_u \cdot \mathbf{h}_{i,s-1} + \mathbf{b}_u), \quad (4b)$$

$$\mathbf{o}_{i,s} = \sigma(W_o \cdot \mathbf{x}_{i,s} + V_o \cdot \mathbf{h}_{i,s-1} + \mathbf{b}_o), \quad (4c)$$

$$\tilde{\mathbf{c}}_{i,s} = \sigma(W_c \cdot \mathbf{x}_{i,s} + V_c \cdot \mathbf{h}_{i,s-1} + \mathbf{b}_c), \quad (4d)$$

$$\mathbf{c}_{i,s} = \mathbf{f}_{i,s} \odot \mathbf{c}_{i,s-1} + \mathbf{u}_{i,s} \odot \tilde{\mathbf{c}}_{i,s}, \quad (4e)$$

$$\mathbf{h}_{i,s} = \mathbf{o}_{i,s} \odot \tanh(\mathbf{c}_{i,s}). \quad (4f)$$

The variables \mathbf{u} , \mathbf{f} , \mathbf{o} and their associated subscripts denote the update, forget, and output gates, respectively. These control gates regulate information to be stored in the cell state $\mathbf{c}_{i,s}$ in (4e) allowing LSTM to achieve long-term memory [11]. The weight matrix for input $\mathbf{x}_{i,s}$ and hidden state $\mathbf{h}_{i,s-1}$ in different gate units are denoted, respectively, by matrices W and V . The variables $\tilde{\mathbf{c}}_{i,s}$ and \mathbf{b} are defined, respectively, as as the cell input activation vector and bias while σ and \tanh are the activation functions. The symbol \odot denotes element-wise multiplication. The ability of a student to achieve the course learning outcomes is therefore encoded in hidden states $\mathbf{h}_{i,s}$, which are then updated when grades for the new courses are made available. Predicted grade \hat{y}_i is then achieved via a fully-connected (FC) layer

$$\hat{y}_i = \mathbf{w} \cdot \mathbf{h}_{i,S}^T + b, \quad (5)$$

where \mathbf{w} and b are defined, respectively, as the weight vector and bias scalar for the FC layer.

2.3 Modeling Relative Performance using GCN

GCN has been applied to model the interactions between nodes in a graph network. As opposed to [12], where the GCN models transitions between courses across semesters, we define \mathbf{A} as the adjacency matrix such that its elements

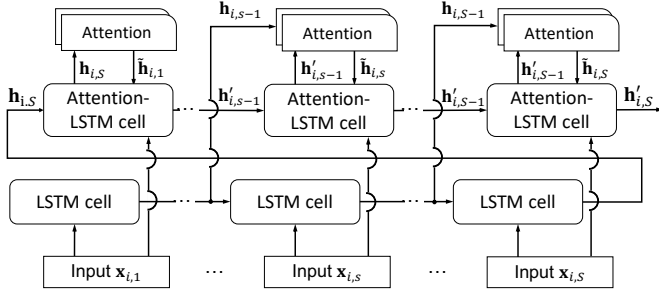


Figure 2: Modeling course importance.

$A_{i,j}$ denotes the similarity of prior grades between two students—a value of 1 is assigned between students i and j having exactly the same prior grades. The model incorporates a feature matrix \mathbf{F} with elements corresponding to first attempt grades that a student obtained over the past semesters for each prior course. Multiple layers of GCNs are then applied to \mathbf{A} and \mathbf{F} with the $(g+1)$ th layer being computed via [20]

$$\mathbf{Z}^{(g+1)} = \sigma \left(\mathbf{D}^{-\frac{1}{2}} \mathbf{A} \mathbf{D}^{-\frac{1}{2}} \mathbf{Z}^{(g)} \mathbf{W}^{(g)} \right). \quad (6)$$

Here, \mathbf{D} is the normalization matrix, $\mathbf{Z}^{(g)}$ is the input to the next layer, and $\mathbf{W}^{(g)}$ is the weight matrix. With the input of the first GCN layer being $\mathbf{Z}^{(0)} = \mathbf{F}$, the output of the last GCN layer is the student-specific graph embedding matrix

$$\mathbf{Z}^{(G)} = [\mathbf{z}_1^T, \dots, \mathbf{z}_K^T]^T, \quad (7)$$

where G is the number of GCN layers. Each node embedding (row) vector \mathbf{z}_i for each node (student) then serves as an input to the subsequent FC layer for grade prediction.

3. THE PROPOSED A2GP MODEL

3.1 Modeling Course Importance

Inspired by the attention mechanism in sequence-to-sequence models [24], the first module comprises an LSTM layer and the attention-LSTM layer to model course importance. To formulate the above and as shown in Figure 2, columns of the grade matrix \mathbf{X}_i in (2) serve as input sequences to the LSTM and the ability to achieve the course learning outcomes is therefore encoded in hidden states $\mathbf{h}_{i,s}$ defined in (4f).

The last hidden state $\mathbf{h}_{i,s}$ will be used to initialize the hidden state of the subsequent attention-LSTM layer. This layer is necessary to account for the influence of the various prior courses on the pilot course of interest. The attention mechanism in this layer can be described by first defining

$$\tilde{\mathbf{h}}_{i,s} = \begin{cases} \mathbf{h}_{i,s}, & \text{if } s = 1; \\ \mathbf{W}_h \cdot [\alpha_{i,s-1} \odot \mathbf{h}_{i,s-1}; \mathbf{h}'_{i,s-1}]^T, & \text{if } s > 1 \end{cases} \quad (8)$$

as the hidden activation vector, where $\mathbf{h}_{i,s-1}$ is the hidden state of the LSTM layer and $\mathbf{h}'_{i,s-1}$ is the hidden state of the attention-LSTM layer. Here, \mathbf{W}_h is the weight matrix that is to be trained, and the prime notation denotes for the attention layer. Therefore, the hidden activation vector $\tilde{\mathbf{h}}_{i,s}$ is

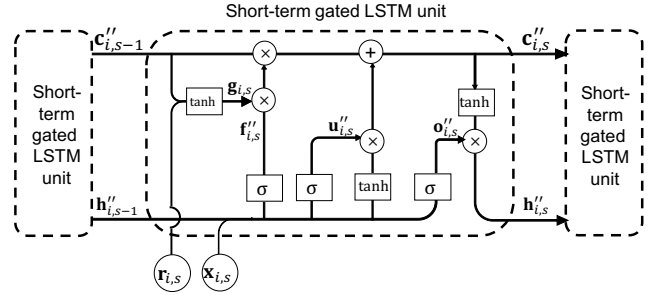


Figure 3: Modeling consistency in student short-term performance.

first initialized as the last hidden state $\mathbf{h}_{i,s}$ from the previous LSTM layer before being updated based on the learned semester-aware attention. Defining \mathbf{W}_α as the weight matrix, the semester-aware attention in (8) for semester s is given by

$$\alpha_{i,s-1} = \text{Softmax}(\mathbf{W}_\alpha [\mathbf{h}_{i,s-1}; \mathbf{h}'_{i,s-1}]). \quad (9)$$

The hidden state for the next unit $\mathbf{h}'_{i,s}$ (of this attention layer) is then computed using student prior grade $\mathbf{x}_{i,s}$ and the hidden activation vector $\tilde{\mathbf{h}}_{i,s}$ such that

$$\mathbf{h}'_{i,s} = \text{Attention-LSTM}(\mathbf{x}_{i,s}, \tilde{\mathbf{h}}_{i,s}), \quad (10)$$

where the Attention-LSTM($\mathbf{x}_{i,s}, \tilde{\mathbf{h}}_{i,s}$) is defined in the same form of LSTM($\mathbf{x}_{i,s}, \mathbf{h}_{i,s}$) in (3) except for the additional attention and hidden activation vector computed in (8) and (9).

We note from (8) that $\tilde{\mathbf{h}}_{i,1}$ is generated from $\mathbf{h}_{i,s}$, which encodes the academic performance over past semesters. This allows the attention-LSTM layer to incorporate both aggregated and semester-based information simultaneously when computing the semester-aware attention for each semester. The last hidden state $\mathbf{h}'_{i,s}$ of the attention-LSTM layer is then used along with short-term performance consistency and the relative performance representation (described below) for the grade prediction.

3.2 Modeling Consistency in Short-term Performance

Modeling short-term variations in academic performance is necessary since such variations may result from active (or the lack of) intervention strategies administered by the academic institution or changes in social-economic status that distracts students away from their academic pursuit [36]. In the second module, we formulate a short-term gated LSTM that employs, for each semester s , the average examination score computed across three consecutive semesters $s-1$, s and $s+1$, i.e.,

$$\mathbf{r}_{i,s} = \begin{cases} [0, \bar{x}_{i,s}, \bar{x}_{i,s+1}], & \text{if } s = 1; \\ [\bar{x}_{i,s-1}, \bar{x}_{i,s}, \bar{x}_{i,s+1}], & \text{if } 1 < s < S; \\ [\bar{x}_{i,s-1}, \bar{x}_{i,s}, 0], & \text{if } s = S, \end{cases} \quad (11)$$

where $\bar{x}_{i,s}$ is the average of non-empty elements in $\mathbf{x}_{i,s}$ defined in (1). Short-term performance averages are then em-

ployed to update information in the memory cell via

$$\mathbf{g}_{i,s} = \tanh(W_g \cdot \mathbf{r}_{i,s} + V_g \cdot \mathbf{c}_{i,s-1} + \mathbf{b}_g), \quad (12a)$$

$$\mathbf{c}_{i,s}'' = \mathbf{g}_{i,s} \odot \mathbf{f}_{i,s}'' \odot \mathbf{c}_{i,s-1}'' + \mathbf{u}_{i,s}'' \odot \tilde{\mathbf{c}}_{i,s}'', \quad (12b)$$

$$\mathbf{h}_{i,s}'' = \mathbf{o}_{i,s}'' \odot \tanh(\mathbf{c}_{i,s}''), \quad (12c)$$

where W_g and V_g are the weight matrices and b_g is the bias term for the short-term gate $\mathbf{g}_{i,s}$. The formulations of $\mathbf{f}_{i,s}''$, $\mathbf{u}_{i,s}''$, $\mathbf{o}_{i,s}''$, the input activation function $\tilde{\mathbf{c}}_{i,s}''$, and the hidden state $\mathbf{h}_{i,s}''$ are identical to those defined in (4). As shown in (12b), both $\mathbf{g}_{i,s}$ and the forget gate $\mathbf{f}_{i,s}''$ control information updates from previous cell state $\mathbf{c}_{i,s-1}''$. The last hidden state $\mathbf{h}_{i,s}''$ is used with other academic achievement representations in the fusion layer for grade prediction.

To gain insights into the above and with reference to Figure 3, the new short-term gate defined in (12a) utilizes $\mathbf{r}_{i,s}$ and $\mathbf{c}_{i,s-1}''$ to determine how short-term consistency in performance affects the cell state update. As opposed to (4a) where the sigmoid function is used, we employ tanh activation in (12a). This is to avoid sharp damp gradients during back propagation, gradient saturation, and gradient updates propagating in different directions when the sigmoid function is used [16, 31]. In addition, $-1 \leq \mathbf{g}_{i,s} \leq 1$ allows the module to model both the positive and negative relationship between performance variation and the cell states.

We also note from (12b) that the previous cell state $\mathbf{c}_{i,s-1}''$ is weighted by both the forget gate $\mathbf{f}_{i,s}$ and the short-term gate $\mathbf{g}_{i,s}$. As per conventional LSTM described by (4a), $\mathbf{f}_{i,s}''$ determines the amount of information to discard from the cell. The short-term gate incorporating $\mathbf{r}_{i,s}$ and $\mathbf{c}_{i,s-1}''$ encapsulates information pertinent to both short-term performance variation and long-term past performance before being passed to the next cell state. Unlike LSTM, where both the input activation $\tilde{\mathbf{c}}_{i,s}''$ and the previous cell state $\mathbf{c}_{i,s-1}''$ are weighted by a gate learned from the current input and the previous hidden state in (4e), only $\mathbf{c}_{i,s-1}''$ is weighted by $\mathbf{g}_{i,s}$ and $\mathbf{f}_{i,s}''$ in (12b). This is because the input gate $\mathbf{u}_{i,s}''$ does not cater for the removal of information from the cell state. Therefore, weighing the input gate $\mathbf{u}_{i,s}''$ by $\mathbf{g}_{i,s}$ in the second term of (12b) is ineffective. Furthermore, applying such weighting on the output gate $\mathbf{o}_{i,s}''$ may result in relevant information being lost in the next hidden state.

3.3 Modeling Relative Performance Against Peers

The third module models the relative performance between students by employing the graph convolutional layer. We first define a $K \times L$ prior score matrix across all students as

$$\mathbf{F} = [\tilde{\mathbf{x}}_1^T, \dots, \tilde{\mathbf{x}}_K^T]^T, \quad (13)$$

where K is the total number of students under consideration and $\tilde{\mathbf{x}}_i = [\tilde{x}_{i,1}, \dots, \tilde{x}_{i,L}]$ with elements $\tilde{x}_{i,l}$ being the first-attempt grade of the i th student for course l . Unlike (2) where grades across every semester are used in the first module, only first attempts are used for the construction of peer-performance graph since they better represent the ability of the student in achieving the learning outcome compared to his/her peers. In addition, the $\{i, j\}$ th element in the pro-

posed $K \times K$ adjacency matrix \mathbf{A} is given by

$$A_{i,j} = \begin{cases} 0, & \text{if } \|\mathbb{N}_{i,j}\| = 0; \\ \rho_{i,j}^{-1} = \left(\frac{\sum_{l \in \mathbb{N}_{i,j}} |\tilde{x}_{i,l} - \tilde{x}_{j,l}|}{\|\mathbb{N}_{i,j}\|} \right)^{-1}, & \text{if } \|\mathbb{N}_{i,j}\| > 0; \\ 1, & \text{if } \tilde{\mathbf{x}}_i = \tilde{\mathbf{x}}_j. \end{cases} \quad (14)$$

Here, we define $\mathbb{N}_{i,j}$ as the set of common courses that students i and j have taken and $\|\mathbb{N}_{i,j}\|$ as the number of such courses. Therefore, $\rho_{i,j}$ denotes the average grade difference that students i and j have achieved for these common courses. The above formulation implies that elements of the adjacency matrix $0 \leq A_{i,j} \leq 1$ correspond to the degree of similarity in academic performance between two students.

With \mathbf{A} and \mathbf{F} , the GCN encodes peer performance via graph representation $\mathbf{Z}^{(g)}$ computed using (6). We apply two GCN layers and the i th row of $\mathbf{Z}^{(2)}$ (denoted by \mathbf{z}_i) constitutes the graph representation corresponding to the relative performance vector for each node (i.e., for the i th student).

3.4 Weighted Fusion Layer and Grade Prediction

To determine the weighting for each academic achievement representation highlighted in Sections 3.1-3.3, a weighted fusion layer is employed. The fusion weight is learned by employing statistics associated with prior and semester average grades. More specifically, we define, for each student i , a 1×4 vector

$$\mathbf{d}_i = [\mu_i, \sigma_i, \mu'_i, \sigma'_i], \quad (15)$$

where μ_i and σ_i are the mean and standard deviation (STD) of non-empty elements in (2). The variables μ'_i and σ'_i denote the mean and STD of

$$\bar{\mathbf{x}}_i = [\bar{x}'_{i,1}, \dots, \bar{x}'_{i,S-1}], \quad (16)$$

where $\bar{x}'_{i,s}$ is the average of non-empty elements across two semesters $\mathbf{x}_{i,s-1}$ and $\mathbf{x}_{i,s}$. We note from the above that \mathbf{d}_i incorporates statistical properties associated with both long- and short-term consistency of a student. These features play an important role in influencing the contribution of each academic achievement representation $\mathbf{h}'_{i,S}$, $\mathbf{h}''_{i,S}$, and \mathbf{z}_i to the predicted grade. This dependence is expected given the student prior grade and the short-term consistency have been used as the input to each module to learn the representations.

To determine the weights in the fusion layer, we first define $p = 1, 2, 3$ as the index for the academic achievement representations. Given \mathbf{d}_i , these weights are learned via an FC layer given by

$$\beta_{i,p} = \mathbf{w}_p \cdot \mathbf{d}_i^T + b_p, \quad (17)$$

where \mathbf{w}_p and b_p are the trainable weight vector and bias for the p th academic achievement representations. The predicted grade \hat{y}_i for student i is then given by

$$\hat{y}_i = \mathbf{w} \cdot [\beta_{i,1} \times \mathbf{h}'_{i,S}; \beta_{i,2} \times \mathbf{h}''_{i,S}; \beta_{i,3} \times \mathbf{z}_i]^T + b, \quad (18)$$

where \mathbf{w} and b are defined, respectively, as the weight vector and bias scalar for the predictor. We employed the mean-

Algorithm 1 The proposed A2GP architecture

Input: Input sequence $X_i = [\mathbf{x}_{i,1}^T, \dots, \mathbf{x}_{i,S}^T]$,
Output: Prediction score \hat{y}_i ,
Module 1: Modeling course importance:
1: **for** student $i \leftarrow 1$ to K **do**
2: **for** $s \leftarrow 1$ to S **do**
3: $\mathbf{h}'_{i,s} \leftarrow$ Attention-LSTM($\mathbf{x}_{i,s}, \tilde{\mathbf{h}}_{i,s}$) using (10)
4: **end for**
5: **end for**
Module 2: Modeling consistency in student short-term performance:
6: **for** student $i \leftarrow 1$ to K **do**
7: **for** $s \leftarrow 1$ to S **do**
8: $\mathbf{r}_{i,s} \leftarrow$ ($\mathbf{x}_{i,s-1}, \mathbf{x}_{i,s}, \mathbf{x}_{i,s+1}$) using (11)
9: $\mathbf{h}''_{i,s} \leftarrow$ Short-term gated LSTM($\mathbf{x}_{i,s}, \mathbf{r}_{i,s}$) using (12)
10: **end for**
11: **end for**
Module 3: Modeling relative performance against peers:
12: $\mathbf{F} = [\tilde{\mathbf{x}}_1^T, \dots, \tilde{\mathbf{x}}_K^T]^T$
13: **for** student $i \leftarrow 1$ to K **do**
14: **for** student $j \leftarrow 1$ to K **do**
15: $A_{i,j} \leftarrow$ ($\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j$) using (14)
16: **end for**
17: **end for**
18: $Z \leftarrow A, F$ using (6)
Module 4: Weighted-fusion:
19: **for** student $i \leftarrow 1$ to K **do**
20: $\mathbf{d}_i = [\mu_i, \sigma_i, \mu'_i, \sigma'_i]$
21: $\beta_{i,p} = \text{FC}(\mathbf{d}_i)$ using (17)
22: $\hat{y}_i \leftarrow \text{FC}([\beta_{i,1} \times \mathbf{h}'_{i,s}, \beta_{i,2} \times \mathbf{h}''_{i,s}, \beta_{i,3} \times \mathbf{z}_{i,s}])$ using (18)
23: **end for**
24: **return** \hat{y}_i

square error loss function

$$\mathcal{L} = \frac{1}{K} \sum_{i=1}^K (y_i - \hat{y}_i)^2 \quad (19)$$

to compute the prediction loss for a total of K students. Similar to [13] [33], a student is classified as at-risk if his/her predicted grade for the pilot course is lower than the pre-defined threshold T , i.e., the classification label is computed by

$$\hat{\varphi}_i = \begin{cases} \text{At-risk,} & \text{if } \hat{y}_i < T; \\ \text{Non at-risk,} & \text{if } \hat{y}_i \geq T. \end{cases} \quad (20)$$

In line with Figure 1 that shows the proposed A2GP architecture, Algorithm 1 provides a formal description of the proposed model.

4. RESULTS AND DISCUSSION

4.1 Datasets

Three datasets have been collected from various departments in a local university with institutional review board (IRB) approval that includes the personal data protection policies. Since other open-source datasets employed for grade prediction do not include past snapshots of examination records, performance of the proposed A2GP model and baseline architectures is evaluated on the datasets from only this university. Seventeen courses across these datasets

Table 1: Number of students in the training and testing set

Department	Course index	Training set		Testing set	
		S	A (%)	S	A (%)
Department 1	C _{1,1}	1241	13.54	249	6.83
	C _{1,2}	2459	1.75	384	1.04
	C _{1,3}	1314	9.97	261	3.83
	C _{1,4}	2448	5.35	357	3.92
	C _{1,5}	1524	7.68	312	3.85
	C _{1,6}	1000	4.70	190	4.74
	Total	9986	6.38	1753	3.76
Department 2	C _{2,1}	1001	12.89	223	8.07
	C _{2,2}	600	12.33	98	7.14
	C _{2,3}	1029	4.76	355	2.25
	C _{2,4}	1034	4.84	205	2.44
	C _{2,5}	987	6.69	167	3.59
	C _{2,6}	1842	7.76	409	3.67
	Total	6493	7.87	1457	4.05
Department 3	C _{3,1}	1001	12.89	133	9.02
	C _{3,2}	600	12.33	84	8.33
	C _{3,3}	1029	4.76	245	3.27
	C _{3,4}	1842	7.76	192	5.21
	C _{3,5}	165	29.70	32	31.25
	Total	4637	9.58	686	6.85

S: number of students

A: percentage of at-risk students

and their corresponding detailed information are illustrated in Table 1. The number of students denoted by ‘‘S’’ and the percentage of at-risk students denoted by ‘‘A’’ in each course for training and testing are also tabulated. These core courses have been offered to all undergraduates across the three engineering departments during their freshman and sophomore years. These datasets include grades obtained by students who are enrolled from academic year (AY) 2015 to 2019. In our context, the pre-defined threshold is $T = 40$ which refers to the passing mark of the university.

To reflect real-world deployment, we predict course grades using examination grades obtained in previous semesters. In particular, the model was trained using prior courses from AY2015 to 2018 and to predict grades for courses registered in AY2019. In this work, the prior grade matrix \mathbf{X}_i is of dimension 20×10 representing twenty prior courses and ten semesters across our dataset. While the undergraduate degree program requires eight semesters to complete, ten semesters were included since some students require a longer duration to graduate.

4.2 Implementation and Performance Metrics

The proposed A2GP model is trained using the Adam optimizer [19]. Hyper-parameters for each model were initialized using the Xavier initialization method [6] and the activation function is a rectified linear unit (ReLU). Adopting a systematic approach [40], the learning rate is first initialized with a small value, e.g., 1×10^{-7} , before being increased exponentially to a pre-defined upper-bound, e.g., 10. An optimal learning rate for the model is then determined from the range in which the model experiences the highest rate of decrease in model loss. This corresponds to 0.8×10^{-3} in our experiments.

Performance of the grade prediction models was evaluated via the mean absolute error defined by

$$\text{MAE} = \frac{1}{N} \sum_{i=1}^N |y_i - \hat{y}_i|, \quad (21)$$

which describes the average absolute error between the predicted and target grades for each grade prediction model. Classification performance of at-risk student detection was evaluated using the F1 score

$$F1 = \frac{2PR}{P + R}, \quad (22)$$

which is computed from the recall (R) and precision (P) scores. The recall score quantifies the number of correct at-risk predictions out of all actual at-risk students in the dataset while precision quantifies the number of correct at-risk predictions out of all detected at-risk students.

4.3 Performance Comparison in Terms of MAE and F1

While many grade prediction algorithms exist, we focus on models that rely only on prior grades obtained from previous semesters. This is important in our context since we cannot assume that all courses are offered online (for the extraction of clickstreams) or that we have access to audio-visual information in a physical classroom setting. To this end, we evaluate the proposed A2GP model on each pilot course by comparing its performance with two widely used classification algorithms (logistic regression (LR) and support vector machine (SVM)), and the LSTM [13]. We have also implemented two variants of the LSTM-based model (attn-LSTM [24] and STG-LSTM) and the GCN [20]. The attn-LSTM was implemented using (8)-(10) while the STG-LSTM is defined by (11)-(12). The GCN was modified from [20] by highlighting the relative performance using (14).

The performance of the models for each department in terms of MAE defined by (21) is tabulated in Table 2. With the predicted grades, at-risk classification performance in terms of the F1 score defined by (22) is tabulated in Table 3 for all predictive models under consideration. Results highlight that the proposed A2GP architecture achieves the lowest average MAE and highest average F1 score than all baseline models across all departments. Compared to the variants of LSTM, the LR and SVM models suffer from poor performance for since these two models do not consider the temporal information that is important for grade prediction.

Table 2: Performance comparison of different models using MAE (a lower value indicates better performance)

Methods	Dept. 1	Dept. 2	Dept. 3	Average
LR	0.121	0.137	0.123	0.127
SVM	0.124	0.138	0.121	0.128
LSTM [13]	0.113	0.129	0.111	0.118
attn-LSTM [24]	0.109	0.128	0.120	0.119
STG-LSTM	0.102	0.124	0.116	0.114
GCN [20]	0.109	0.121	0.115	0.115
The proposed A2GP model	0.104	0.119	0.109	0.111

Dept.: Department

It is also interesting to note that all models achieve lower grade prediction and at-risk classification performance for Departments 2 and 3 compared to Department 1. This is because Department 1 has a mandatory set of courses for all students while students from Departments 2 and 3 have the freedom to select courses not offered by their respective schools. Due to this difference in the course selection procedure, there are fewer commonly taken courses between students in Departments 2 and 3. Since GCN determines the relative performance of a student in comparison to his/her peers, an insufficient number of overlapping courses makes it more challenging to predict the grades for the pilot courses. On the same note, since there exist various combinations of courses taken by students from Departments 2 and 3, the LSTM-based models (attn-LSTM and STG-LSTM) are unable to identify similar sequences of temporal information among students. Therefore, diversity in the prior courses results in the poor prediction of a common pilot course grade.

In terms of at-risk classification for Department 1, the attn-LSTM model (with an F1 score of 0.324) achieves an approximate 17% improvement compared to LSTM. This improvement is attributed to attn-LSTM being able to detect the prior courses with higher importance compared to LSTM that equally weighs all courses to determine the grade for the pilot courses. The STG-LSTM model, on the other hand, accounts for the short-term fluctuation of student performance during the update of hidden states. This is in contrast to LSTM that updates the hidden state with equal importance applied to previous hidden states irrespective of any variations in performance. This results in a 10% reduction in the average MAE and 30% increase in an average F1 score for STG-LSTM over that of LSTM.

With regard to the non-temporal approach, since the GCN model constructs an input graph based on grade differences between student pairs (for identifying relative performance), variance of grades within a given dataset would determine the extent of distinguishability among the students. Figure 4 shows the relationship between the F1 score obtained using GCN and the standard deviation σ_{C_i} of the prior grades for

Table 3: At-risk prediction performance using F1 score (a higher value signifies better performance)

Department index	Course index	LR	SVM	LSTM [13]	attn-LSTM [24]	STG-LSTM	GCN [20]	The proposed A2GP model
Department 1	C _{1,1}	0.372	0.341	0.293	0.4	0.457	0.375	0.439
	C _{1,2}	0.222	0.222	0.24	0.308	0.4	0.286	0.444
	C _{1,3}	0.345	0.222	0.348	0.381	0.438	0.308	0.48
	C _{1,4}	0.174	0.154	0.34	0.218	0.381	0.211	0.368
	C _{1,5}	0.313	0.294	0.25	0.435	0.467	0.313	0.526
	C _{1,6}	0.143	0.222	0.143	0.2	0.154	0	0.286
	Average		0.261	0.243	0.269	0.324	0.383	0.249
Department 2	C _{2,1}	0.291	0.314	0.241	0.321	0.290	0.276	0.328
	C _{2,2}	0.235	0.264	0.3	0.174	0.25	0.111	0.348
	C _{2,3}	0.286	0.264	0.318	0.273	0.19	0.231	0.32
	C _{2,4}	0	0	0.231	0.095	0.286	0	0.08
	C _{2,5}	0	0.167	0.182	0.222	0.133	0	0.273
	C _{2,6}	0.313	0.3	0.341	0.114	0.217	0.211	0.28
	Average		0.187	0.218	0.269	0.200	0.228	0.138
Department 3	C _{3,1}	0.214	0.244	0.267	0.353	0.357	0.308	0.4
	C _{3,2}	0.3	0.221	0.353	0.308	0.174	0.25	0.222
	C _{3,3}	0.253	0.244	0.353	0.154	0.4	0.333	0.4
	C _{3,4}	0.1	0.164	0.235	0.118	0.077	0	0.167
	C _{3,5}	0.221	0.2	0.571	0.615	0.621	0.444	0.606
	Average		0.218	0.215	0.356	0.310	0.326	0.267
Average		0.222	0.226	0.295	0.276	0.311	0.215	0.351

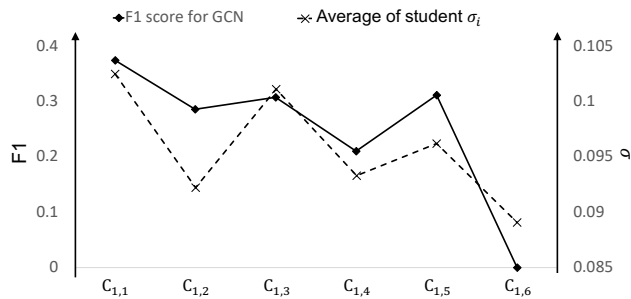


Figure 4: Relationship between F1 score of modeling relative performance against peer and average of σ_i .

all students. A smaller value of σ_{C_i} denotes a high similarity in performance among all students taking that course, implying that it is challenging for GCN to differentiate at-risk student performance from well-performing students (low F1 score). Therefore, we note that the ability of GCN for grade prediction is dependent on the underlying statistical properties of a dataset, resulting in varying performance across the courses for Department 1. We also note that GCN is not able to detect any at-risk students for some of the courses (reflected by the F1 score of zero in Table 1) since there were only few students who were actually at-risk. With most of the students achieving good grades in these courses, the performance variation among students is minimal, resulting in GCN not being able to identify relative performance differences.

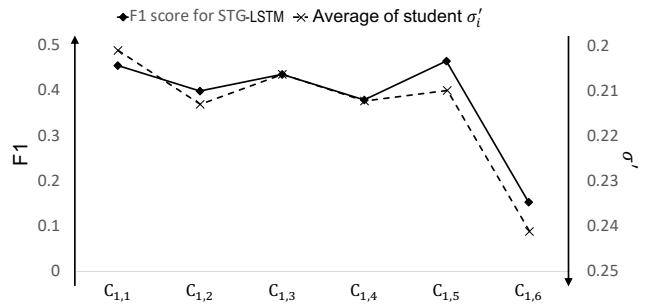


Figure 5: Relationship between F1 score of modeling short-term performance and average of σ'_i .

Compared to the above models, the proposed A2GP model achieves the lowest average MAE of 0.111 and highest average F1 score of 0.351 across the three departments. These results highlight the importance of synthesizing the three dimensions associated with course importance, performance consistency, and benchmarking. Although the performance of individual modules (attention module and short-term gated module) are modestly higher in comparison to LSTM, A2GP includes a weighted fusion that adaptively determines the importance of each module depending on the relevance of the academic achievement representations to each dataset. Figure 6 illustrates the performance of our proposed A2GP model in terms of F1 score with a weighted fusion layer (implemented via (18)) or an equal fusion layer (where $\beta_1 = \beta_2 = \beta_3 = 1$). It is important to highlight that the weighted

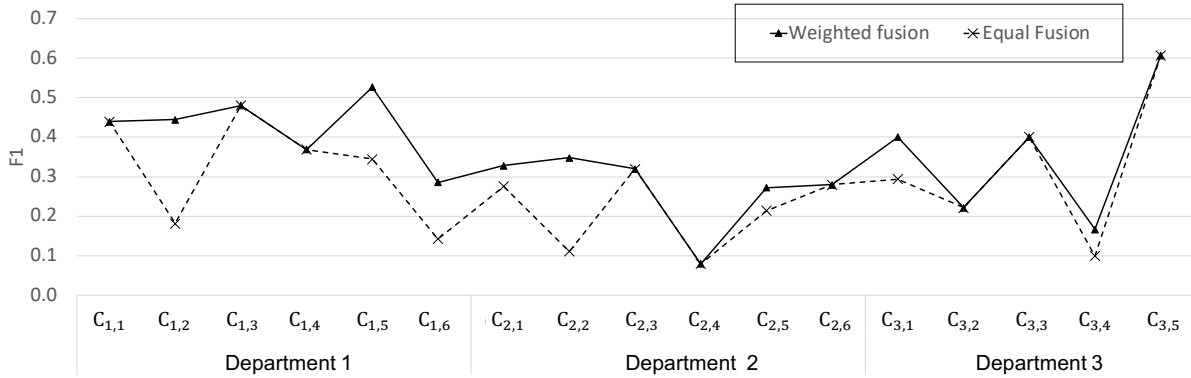


Figure 6: F1 score across courses for model with/without weighted fusion.

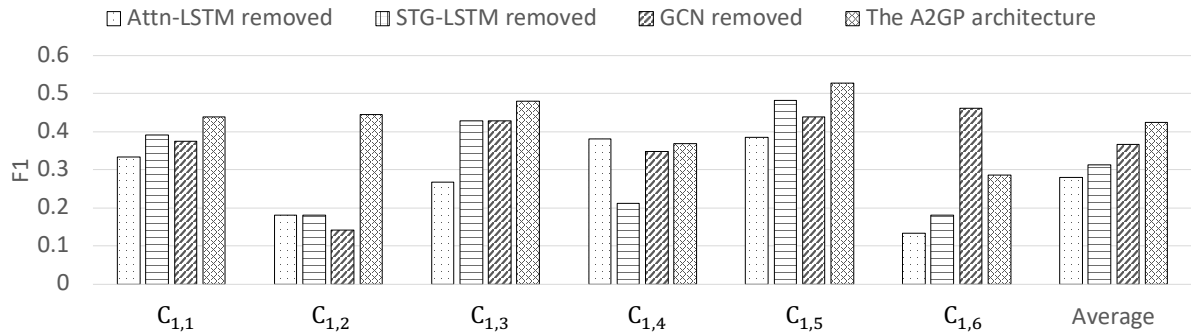


Figure 7: Ablation of the proposed architecture.

fusion layer achieves significantly higher F1 scores for several courses with the remaining courses exhibiting similar performance as that of the equally weighted configuration. In particular, A2GP with weighted fusion (implemented via (17)) achieves an improvement of 0.263 F1 score for course $C_{1,2}$ over that of the equally fusion strategy.

As described in Section 3.4, the underlying statistics of student prior exam grades \mathbf{d}_i have been used to learn the fusion weights $\beta_{i,p}$. Figure 5 shows the variation of F1 with the mean of STD σ' defined by averaging σ'_i according to (15) over all students. Results plotted in this figure was generated by evaluating STG-LSTM over all courses offered by Department 1. A high value of σ' implies many students in this course exhibit short-term performance fluctuations in the past semesters. It can be seen that the F1 score reduces with increasing σ' . This implies that higher fluctuations in short-term student performance will pose a challenge for the model to detect at-risk students. To address limitations faced by individual models, the weighted fusion layer in the A2GP model de-emphasizes the aspects affected by the dataset while emphasizing other academic achievement representations which, in turn, aid the grade prediction process.

4.4 Ablation Test

Figure 7 shows results associated with an ablation test performed on the A2GP model. Here, at-risk prediction per-

formance is determined when each of the academic achievement representation is removed from the A2GP architecture. Among all three representations, the average performance across all courses reduces most significantly when attn-LSTM, i.e., $\mathbf{h}'_{i,s}$ is removed. This is due to the impact of identifying course importance on the entire cohort since constructivist approach is often adopted during curriculum development. The STG-LSTM model, on the other hand, is student-specific—fluctuation in individual performance depends on unique circumstances not generalizable for the other students. Therefore, the A2GP model is less sensitive to STG-LSTM compared to attn-LSTM. Removing GCN results in the least difference in prediction performance since different courses exhibit varying degree of grade spread, with some lacking sufficient information for GCN to discern among student performance resulting in an inappropriate representation. Nonetheless, incorporating peer performance will still be beneficial to A2GP model as seen in Figure 7.

5. CONCLUSIONS AND FUTURE WORK

An academic achievement-based grade prediction architecture is proposed for grade prediction and at-risk student detection. To utilize three important aspects in student prior performance—course importance, short-term performance fluctuation, and relative performance against peers, three modules have been formulated and fused. The first module learns the prior grade representation along with course

importance by employing the attention-based LSTM model. The new STG-LSTM in the second module is motivated by the need to model short-term fluctuation in academic performance. The third module is motivated by the need to model relative performance when detecting at-risk students—students are often deemed as at-risk if their performance is consistently below par compared to their peers. We evaluated the prediction performance of the proposed A2GP model by comparing its performance with baseline models. Results obtained showed that the proposed architecture outperforms existing at-risk detection algorithms across seventeen undergraduate courses from three departments. Improving the F1 score in at-risk student detection facilitates the administration of pre-emptive interventions by instructors, counsellors, or pastoral care managers.

There are possible avenues for future work. First, A2GP has been optimized for use with prior examination grades and has not been validated for online learning activities. Behavioral features associated with the consumption of online assets may provide additional (and often complementary) information that may aid grade prediction. Secondly, further investigations can be performed on the fusion weights defined by (17). These weights may offer insights into the importance of different representations influenced by characteristics that govern the cohort performance within each course.

6. REFERENCES

- [1] S. Adak. Effectiveness of constructivist approach on academic achievement in science at secondary level. *Edu. Research Reviews*, 12(22):1074–1079, 2017.
- [2] M. F. Ayaz and H. Şekerci. The effects of the constructivist learning approach on student’s academic achievement: A meta-analysis study. *Turkish Online J. Edu. Tech.*, 14(4):143–156, 2015.
- [3] W. B. Chik. The Singapore personal data protection act and an assessment of future trends in data privacy reform. *Comput. Law & Security Review*, 29(5):554–575, 2013.
- [4] R. Dattakumar and R. Jagadeesh. A review of literature on benchmarking. *Benchmarking: Int. J.*, 10(3):176–209, 2003.
- [5] E. Fincham, B. Roozemberczki, V. Kovanovic, S. Joksimovic, J. Jovanovic, and D. Gasevic. Persistence and performance in co-enrollment network embeddings: An empirical validation of Tinto’s student integration model. *IEEE Trans. Learn. Tech.*, 14(1):106–121, 2021.
- [6] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proc. Int. Conf. Artif. Intell. and Statist.*, pages 249–256, 2010.
- [7] N. Granott. How microdevelopment creates macrodevelopment: Reiterated sequences, backward transitions, and the zone of current development. *Microdevelopment: Transition Processes in Development and Learn.*, 7:213–244, 2002.
- [8] Q. Guo, M. Chen, D. An, and L. Ye. Prediction of students’ course failure based on campus card data. In *Proc. IEEE Int. Conf. Robots & Intell. Sys.*, pages 361–364, 2019.
- [9] T. Gutenbrunner, D. D. Leeds, S. Ross, M. Riad-Zaky, and G. M. Weiss. Measuring the academic impact of course sequencing using student grade data. In *Proc. Int. Conf. Educational Data Mining*, pages 1–5, 2021.
- [10] J. M. Harackiewicz, K. E. Barron, J. M. Tauer, S. M. Carter, and A. J. Elliot. Short-term and long-term consequences of achievement goals: Predicting interest and performance over time. *J. Edu. Psych.*, 92(2):316–330, 2000.
- [11] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.
- [12] Q. Hu and H. Rangwala. Academic performance estimation with attention-based graph convolutional networks. In *Proc. Int. Conf. Educational Data Mining*, pages 69–78, 2019.
- [13] Q. Hu and H. Rangwala. Reliable deep grade prediction with uncertainty estimation. In *Proc. Int. Conf. Learn. Anal. & Knowl.*, pages 76–85, 2019.
- [14] N. Huntington-Klein and A. Gill. Semester course load and student performance. *Research in Higher Educ.*, 62(5):623–650, 2021.
- [15] Z. Iqbal, A. Qayyum, S. Latif, and J. Qadir. Early student grade prediction: an empirical study. In *Proc. IEEE Int. Conf. Advancements in Computational Sci.*, pages 1–7, 2019.
- [16] B. L. Kalman and S. C. Kwasny. Why tanh: choosing a sigmoidal function. In *Proc. IEEE Int. Joint Conf. Neural Netw.*, volume 4, pages 578–581, 1992.
- [17] B.-H. Kim, E. Vizitei, and V. Ganapathi. GritNet: Student performance prediction with deep learning. In *Proc. Int. Conf. Educ. Data Mining*, pages 625–629, 2018.
- [18] J. S. Kim. The effects of a constructivist teaching approach on student academic achievement, self-concept, and learning strategies. *Asia Pacific Edu. Review*, 6:7–19, 2005.
- [19] D. Kingma and J. Ba. Adam: A method for stochastic optimization. 2015.
- [20] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *Proc. Int. Conf. Learn. Representations*, pages 76–85, 2017.
- [21] I. Lee, D. Kim, S. Kang, and S. Lee. Ensemble deep learning for skeleton-based action recognition using temporal sliding LSTM networks. In *Proc. IEEE Int. Conf. Comput. Vis.*, pages 1012–1020, 2017.
- [22] K. Liu, S. Tatinati, and A. W. Khong. Context-based data model for effective real-time learning analytics. *IEEE Trans. Learn. Technologies*, 13(4):790–803, 2020.
- [23] X. Lu, Y. Zhu, Y. Xu, and J. Yu. Learning from multiple dynamic graphs of student and course interactions for student grade predictions. *Neurocomputing*, 431:23–33, 2021.
- [24] T. Luong, H. Pham, and C. D. Manning. Effective approaches to attention-based neural machine translation. In *Proc. Int. Conf. Empirical Methods in Natural Lang. Process.*, pages 1412–1421, 2015.
- [25] M. Marschark, D. M. Shaver, K. M. Nagle, and L. A. Newman. Predicting the academic achievement of deaf and hard-of-hearing students from individual, household, communication, and educational factors. *Exceptional Children*, 81(3):350–369, 2015.

- [26] H. R. Mkwazu and C. Yan. Grade prediction method for university course selection based on decision tree. In *Proc. Int. Conf. Aviation Safety Info. Tech.*, pages 593–599, 2020.
- [27] R. Murata, A. Shimada, and T. Minematsu. Early detection of at-risk students based on knowledge distillation rnn models. In *Proc. Int. Conf. Educational Data Mining*, pages 1–5, 2021.
- [28] K. H. R. Ng, S. Tatinati, and A. W. H. Khong. Online education evaluation for signal processing course through student learning pathways. In *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Process.*, pages 6458–6462, 2018.
- [29] K. H. R. Ng, S. Tatinati, and A. W. H. Khong. Grade prediction from multi-valued click-stream traces via Bayesian-regularized deep neural networks. *IEEE Trans. Signal Process.*, 69:1477–1491, 2021.
- [30] K. Niu, X. Cao, and Y. Yu. Explainable student performance prediction with personalized attention for explaining why a student fails. In *Proc. Int. Assoc. Advancement Artificial Intell. Workshop AI Edu.*, pages 1–11, 2021.
- [31] C. E. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall. Activation functions: comparison of trends in practice and research for deep learning. In *Proc. Int. Conf. Computational Sci. and Tech.*, pages 124–133, 2021.
- [32] F. Okubo, T. Yamashita, A. Shimada, and H. Ogata. A neural network approach for students’ performance prediction. In *Proc. Int. Learn. Anal. & Knowl. Conf.*, pages 598–599, 2017.
- [33] A. P. Patil, K. Ganesan, and A. Kanavalli. Effective deep learning model to predict student grade point averages. In *Proc. IEEE Int. Conf. Computational Intell. and Computing Research*, pages 1–6, 2017.
- [34] A. Polyzou and G. Karypis. Grade prediction with course and student specific models. In *Proc. Pacific-Asia Conf. Knowl. Discovery and Data Mining*, pages 89–101. Springer, 2016.
- [35] A. Polyzou and G. Karypis. Grade prediction with models specific to students and courses. *Int. J. Data Sci. and Anal.*, 2(3):159–171, 2016.
- [36] K. Rajandran, T. C. Hee, S. Kanawarthy, L. K. Soon, H. Kamaludin, and D. Khezrimotlagh. Factors affecting first year undergraduate students academic performance. *Scholars J. of Economics, Business, and Management*, 2(1A):54–60, 2015.
- [37] S. H. Seyyedrezaie and G. Barani. Constructivism and curriculum development. *J. Humanities Insights*, 1(3):119–124, 2017.
- [38] A. M. Shahiri and W. Husain. A review on predicting student’s performance using data mining techniques. *Procedia Comput. Sci.*, 72:414–422, 2015.
- [39] G. Siemens and D. Gašević. Special issue on learning and knowledge analytics. *Edu. Technol. & Society*, 15(3):1–163, 2012.
- [40] L. N. Smith. Cyclical learning rates for training neural networks. In *Proc. IEEE Winter Conf. Appl. of Comput. Vis.*, pages 464–472, 2017.
- [41] R. R. Subramanian, D. V. V. S. S. S. Babu, D. U. Rani, M. Devendrareddy, B. Kusuma, and R. R. Sudharsan. Detecting bias in the relative grading system using machine learning. In *Proc. Int. Conf. Advancements Electrical Electronics Comm. Comp. Automation (ICAECA)*, pages 1–6, 2021.
- [42] S. Syme, C. Davis, and C. Cook. Benchmarking australian enabling programmes: assuring quality, comparability and transparency. *Assessment Eval. Higher Edu.*, 46(4):572–585, 2020.
- [43] H. J. Walberg, B. J. Fraser, and W. W. Welch. A test of a model of educational productivity among senior high school students. *J. Edu. Research*, 79:133–139, 1986.
- [44] J. Whitehill, K. Mohan, D. Seaton, Y. Rosen, and D. Tingley. MOOC dropout prediction: How to measure accuracy? In *Proc. ACM Conf. Learn. and Scale*, pages 161–164, 2017.
- [45] A. T. Widjaja, L. Wang, T. T. Nghia, A. Gunawan, and E.-P. Lim. Next-term grade prediction: A machine learning approach. In *Proc. Int. Conf. Educational Data Mining*, pages 700–703, 2020.
- [46] T. Yang, C. Brinton, C. Joe-Wong, and M. Chiang. Behavior-based grade prediction for MOOCs via time series neural networks. *IEEE J. Sel. Topics in Signal Process.*, 11(5):716–728, 2017.
- [47] Y. Zhang, R. An, S. Liu, J. Cui, and X. Shang. Predicting and understanding student learning performance using multi-source sparse attention convolutional neural networks. *IEEE Trans. Big Data*, Early Access:1–14, 2021.

Individual Fairness Evaluation for Automated Essay Scoring System

Afrizal Doewes
Eindhoven University of
Technology,
The Netherlands
a.doewes@tue.nl

Akrati Saxena
Eindhoven University of
Technology,
The Netherlands
a.saxena@tue.nl

Yulong Pei
Eindhoven University of
Technology,
The Netherlands
y.pei.1@tue.nl

Mykola Pechenizkiy
Eindhoven University of
Technology,
The Netherlands
m.pechenizkiy@tue.nl

ABSTRACT

In Automated Essay Scoring (AES) systems, many previous works have studied group fairness using the demographic features of essay writers. However, individual fairness also plays an important role in fair evaluation and has not been yet explored. Initialized by Dwork et al. [10], the fundamental concept of individual fairness is “similar people should get similar treatment”. In the context of AES, individual fairness means that “similar essays should be treated similarly”. In this work, we propose a methodology to measure individual fairness in AES. The similarity of essays can be computed using the distance of the text representation of essays. We compare several text representations of essays, from the classical text features, such as BOW and TF-IDF, to the more recent deep-learning-based features, such as Sentence-BERT and LASER. We also show their performance against paraphrased essays to understand if they can maintain the ranking of similarities between the original and the paraphrased essays. Finally, we demonstrate how to evaluate the performance of automated scoring systems models with regard to individual fairness by counting the number of pairs of essays that satisfy the individual fairness equation and by observing the correlation of score difference with the distance of essays. Our analysis suggests that the Sentence-BERT, as the text representation of the essays, and Gradient Boosting, as the score prediction model, provide better results based on the proposed individual fairness evaluation methodology.

Keywords

Individual Fairness, Automated Essay Scoring System, Distance Metric, Rank Evaluation

A. Doewes, A. Saxena, Y. Pei, and M. Pechenizkiy. Individual fairness evaluation for automated essay scoring system. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 206–216, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853151>

1. INTRODUCTION

Automated Essay Scoring (AES) systems have been widely used in a vast number of educational institutions [25, 26]. However, there is a growing concern about the fairness issues in such a system so that no student should be treated unfairly [4, 5, 7, 16]. In recent years, many research studies have explored group fairness in AES with respect to demographic features, gender, race, socioeconomic status, and nationality of the writers [16]. Bridgeman et al. [4] studied the fairness of the essay scoring tool e-rater, and observed that in some cases, human and e-rater scores were incomparable across some subgroups. In particular, students’ essays from China tend to get higher scores from e-rater than from humans. The authors also mentioned that the rankings of subgroups would be different if machine scores were substituted for human scores. Additionally, Burstein and Chodorow [5] evaluated essays using the Test of Written English (TWE) and found that Arabic and Spanish speakers received relatively higher scores from humans, while Chinese speakers received higher scores from the machine. Although the differences between various ethnic groups are generally not large, they are still notable in some circumstances. Another study on the fairness of essay scoring system from the Analytical Writing Assessment to six subpopulation groups of GMAT[®] test takers was conducted in [12]. On the contrary, this study found that none of the subpopulation groups has an unfair advantage and is unfairly punished by the system. In addition, most of the work on automated essay scoring focuses on the effort of maximizing the agreement with the human raters, although human scores, which are treated as the gold standard for training a machine learning model, are not free from bias [1]. For group fairness, the availability of demographic features of the students is mandatory. However, privacy and legal concerns prevent the students’ personal information from being published, which is the main limitation in such works.

Besides group fairness, maintaining individual fairness for each graded essay is also important in AES systems. Initialized by Dwork et al. [10], the fundamental concept of individual fairness is that “similar people should get similar treatment”. In the context of AES, this concept is transformed into “similar essays should get similar treatment”.

To study individual fairness extensively in AES, we propose a methodology to assess the performance of automated scoring systems with regard to individual fairness. In our work, the similarity and dissimilarity of essays are computed using the distance metrics on the textual feature representation of essays. We investigate the performance of several vector representations of the essays, from the classical text features, such as Bag-of-Words (BOW) and Term Frequency - Inverse Document Frequency (TF-IDF), to the more recent deep-learning-based features, such as Sentence-BERT [21] and Language Agnostic Sentence Representation (LASER) [2]. However, these text representations of the essays are numerical vectors, and it is not easy for humans to interpret such results. Therefore, we highlight the importance of defining interpretable essay features, such as answer length, unique word count, spelling errors, and language errors, to help humans judge whether pairs of essays are similar or not.

The similarity score computed using the distance metric on the textual feature representation of the essays assigns a numeric value to each pair of essays, and the number of pairs can be very large. Consider if our dataset contains 1500 data points, then there will be 1,124,250 unique pairs. It makes the job of human evaluators in judging the similarity of each pair practically impossible. Therefore, we present two ways to compare how the distance metrics perform in measuring the essay similarities in the presence of this large number of pairs: firstly, by examining the extreme cases (most similar and most dissimilar essays), and secondly, by using aggregates of features differences. We further assess the quality of the essay vector representations and the distance metric by evaluating their performance against paraphrased essays.

Finally, we assess the performance of automated essay scoring models with regard to individual fairness using the proposed methodology. Firstly, we calculate the percentage of the number of pairs that satisfy the individual fairness equation (described in subsection 5.3.2) and observe that different text features have different distance distributions that affect the fairness results. Secondly, we categorize the essays based on the distance metric and visualize the score difference between different categories of essays. We observe that the score difference of more similar pairs is lower than the score difference of the less similar ones, as expected.

To the best of our knowledge, this is the first work that addresses individual fairness in automated essay scoring systems. The rest of the paper is organized as follows. In Section 2, we discuss the problem statement. In Section 3, we discuss the individual fairness assessment methodology for AES. Sections 4 and 5 cover the experimental settings and evaluation results, respectively. The paper is concluded in the last section with future directions.

2. PROBLEM STATEMENT

In this work, we aim to evaluate individual fairness in Automated Essay Scoring (AES) systems. Individual fairness in AES means that “similar essays should be treated similarly”, which raises several fundamental questions as follows.

1. How to define similar essays?
 - (a) How to measure similarity between essays? If the

similarity is computed using distance metrics on the text representation of essays, which text representation is good enough and which distance metric should be used?

- (b) How to make the evaluation of the distance metric interpretable for humans?
 - (c) Are there any limitations to the text representations of the essays?
2. How to define individual fairness in AES?
 - (a) How to define individual fairness in the presence of a large number of pairs of essays?
 - (b) Can we define individual fairness using the interpretable features of essays?
 3. How to evaluate individual fairness using essays’ similarity? Which machine learning model works best for maintaining individual fairness in essays evaluation?

3. INDIVIDUAL FAIRNESS ASSESSMENT METHODOLOGY FOR AES

To evaluate individual fairness in AES, we propose an end-to-end methodology from (i) selecting the text representation of essays to define the similarity, (ii) evaluating the performance of distance metrics, to (iii) evaluating the performance of the regression models with regard to individual fairness.

1. **Text Representation Selection.** To process the essays using a machine learning model, we first need to represent the essays in a vector format. In the field of Natural Language Processing (NLP), there have been proposed many methods to transform a text into a numeric vector, also known as the vector representation of the text. In this paper, we analyze several vector representations of essays, from the classical text feature, such as Bag-of-Words (BOW) and Term Frequency-Inverse Document Frequency (TF-IDF), to the more recent deep-learning-based extracted features, such as Sentence-BERT and LASER.
2. **Distance Metrics Selection.** There are many ways to calculate the distance of two vectors, and the most common distance metrics for textual data are cosine distance, Euclidean distance, Manhattan distance, and Jaccard distance. In our study, we choose cosine distance since it performs well for textual data, as shown in previous NLP studies [23, 6, 20].
3. **Evaluate Essay Similarity.** To evaluate the similarity of essays using distance metrics, we propose the following methods.
 - (a) **Define Interpretable Essay Features.** The text representation of an essay, generated using text representation API, provides a numeric vector that is not interpretable for humans. However, it is important that humans can also compare whether two essays are similar or not using interpretable features. Therefore, we define the interpretable essay features, which are discussed in detail in subsection 5.1.1.

- (b) **Extreme Cases.** In the presence of a large number of pairs, it is an impossible task for humans to examine each pair one by one. Therefore, we suggest a method to get a quick understanding of the performance of the distance metrics using the extreme cases based on the distance of essays. In this context, extreme cases are the most similar and the most dissimilar pair of essays. We also visualize the similarity (or dissimilarity) of a pair based on the interpretable essay features.
 - (c) **Aggregate of Features Differences.** Analyzing only the extreme cases is insufficient to make a concrete conclusion, as we can not judge the performance of the distance metrics of millions of pairs of essays only based on the nearest and farthest pairs. Therefore, to examine all of the essay pairs, we use aggregated feature differences computed using the normalized feature values. In this paper, we sampled 30,000 pairs of essays from three different similarity groups; lowest distant pairs (most similar), medium distant pairs (moderate similarity), and farthest distant pairs (most dissimilar). In each group, we average the score difference of 10,000 pairs of essays and compare this average score difference of all three groups. We visualize the average difference for different features that also help in understanding the dataset; more details are provided in Section 5.1.3.
4. **Evaluate the Distance Metric against Paraphrased Essays.** We further examine whether the text representation of the essays can recognize paraphrased essays and maintain their similarity ranking, as the original essay as well as its corresponding paraphrased essay should be treated similarly by an individual-fair AES system. To simulate this process, we compare the distance metric of the original essays and the paraphrased essays, and compare their similarity ranking. We measure the ranking quality using the Normalized Discounted Cumulative Gain (NDCG) score [13].
5. **Evaluate Individual Fairness.** The next step is to evaluate the performance of the scoring models with regard to the individual fairness measure. Individual fairness requires that the distance between two individual essays' outcomes is no greater than their distance according to the distance metric.
- (a) **Based on the Individual Fairness Equation.** We count the number of pairs that satisfy the *Lipschitz mapping* function by Dwork et al. [10] and represent what percentage of essays follows it.
 - (b) **Score Difference vs. Distance of Essays.** We study the correlation of score difference versus the distance of essays based on their text representation, as this is a quick method to check whether similar essays are treated similarly or not. If the system is fair, then the score difference between essays must correlate with their distance in the vector space. The average score difference in the group of similar essays should also be lower than the average score difference in the group of dissimilar essays. The expected condition from the visualization is a left-triangle-shaped (linear increment

in score difference as the distance of essays increases) graph for each scoring model.

4. EXPERIMENTAL SETUP

In this section, we introduce the experimental setup used in our work. We discuss the dataset, text representation of essays using several text encoding techniques in NLP, and regression models to predict the score of the essays that we use for the evaluation.

4.1 Dataset

We use the Automated Student Assessment Prize (ASAP) dataset¹, hosted by the Kaggle platform, and use the dataset of prompt 7. These essays were written by Grade-7 students. The total number of essays in this prompt is 1569 and had an average length of 187 words. The students were asked to write a story about a time when they were patient or about a time when someone they knew was patient or write a story in their own way about patience. Their answers were graded according to four rubrics: (i) ideas, (ii) organization, (iii) style, and (iv) conventions. Each essay was scored by two human raters on a 0 - 3 integer scale. Final scores were resolved by adding the rubric scores assigned by the human raters, resulting in a resolved rubric score between 0 and 6.

4.2 Text Representation of Essays

We use the following text representation methods for generating the vector feature representation of essays.

4.2.1 Bag-of-Words (BOW)

We first evaluate the performance of the BOW representation of essays in measuring the similarities between them since BOW is one of the classical textual features that has been widely used in NLP. BOW vector is constructed using a set of n-grams from the text. The n-gram is a contiguous sequence of n words from a given text or speech and is extensively used in NLP tasks. Specifically, n-grams are named *unigram*, *bigram*, and *trigram* when n is set to be 1, 2, and 3, respectively. In this paper, we generate our BOW features using unigram and bigram. For example, from a sentence 'Patience is important', the resulting n-grams are 'patience', 'is', 'important', 'patience is', and 'is important'.

Before creating n-grams, we implemented some text preprocessing techniques on the essays. We removed all non-letter characters and lemmatized the words using WordNet Lemmatizer from NLTK (Natural Language Toolkit) python library [3]. While building the vocabulary for n-grams, we ignored the terms that appear in less than three documents to remove the infrequently used terms. As a result, for each essay, we obtain a feature vector with a length of 14,974.

4.2.2 Term Frequency-Inverse Document Frequency (TF-IDF)

TF-IDF is a statistical measure that evaluates how relevant a word is to a document in a collection of documents. TF-IDF consists of two components, (i) Term Frequency (TF) that measures how frequently a term occurs in a document, and (ii) Inverse Document Frequency (IDF) that diminishes

¹<https://www.kaggle.com/c/asap-aes>

the weight of terms that occur very frequently in the document set and increases the weight of terms that occur rarely. Formally, TF-IDF is defined as follows.

$$tfidf_{(t,d)} = tf_{(t,d)} \times idf_{(t)} \quad (1)$$

where $tf_{(t,d)}$ is the number of occurrence of term t in document d , and according to the scikit-learn documentation, the $idf_{(t)}$ is computed as,

$$idf_{(t)} = \log \left(\frac{1+n}{1+df_{(t)}} \right) + 1 \quad (2)$$

where n is the total number of documents and $df_{(t)}$ is the number of documents that contain term t .

The resulting TF-IDF vectors are then normalized by the Euclidean norm. The effect of adding “1” to the IDF in the equation above is that terms with zero IDF, i.e., terms that occur in all documents, will not be completely ignored. Furthermore, the constant “1” is added to the numerator and denominator of the IDF as if an extra document was seen containing every term in the dataset exactly once, which prevents zero divisions. Before creating the TF-IDF vector, we applied the same text preprocessing techniques on the essays as the Bag-of-Words (BOW) vector. To create the TF-IDF vector, we use the scikit-learn library of TfidfVectorizer [18].

4.2.3 Sentence-BERT

We enrich our experiments by including more recent deep-learning-based features, besides the aforementioned classical text features. Sentence-BERT, which is introduced by Reimers and Gurevych (2019), is a modification of the pre-trained BERT [14] network using Siamese and Triplet network [21]. It converts a sentence into a 768-dimension feature vector and produces semantically meaningful sentence embedding. The embedding results can be used for similarity calculation. The Sentence-BERT representation has been proven to be able to separate the topics in the ASAP dataset very well [9].

4.2.4 Language-Agnostic Sentence Representations (LASER)

LASER was released by Facebook to support multilingual sentence representations to carry out various NLP tasks. They provide an encoder that was trained on more than 90 languages, written in 28 different scripts [2]. LASER includes all European languages, many Asian, Indian, Arabic, and Persian languages, along with numerous minority languages and dialects. All these languages are encoded by the same BiLSTM encoder with a shared BPE (Byte-Pair Encoding) vocabulary. Compared to Sentence-BERT, which produces a 768 length vector, LASER generates a 1024 dimensional sentence vector representation. The pre-trained LASER model is available online².

4.3 Regression Models & Evaluation Measures

We trained three regression models to predict essay scores: (i) Random Forest, (ii) Gradient Boosting, and (iii) Ridge Regression. We split the dataset into 80% training data and

²<https://github.com/facebookresearch/LASER>

Table 1: Essay Features

No.	Interpretable Essay Features
1	Answer Length (Character counts)
2	Word count
3	Average word length
4	Count of “good” POS n-grams
5	Number of overlapping tokens with the prompt
6	Number of overlapping tokens (including synonyms) with the prompt
7	Number of punctuation
8	Spelling errors
9	Unique words count
10	Prompt – answer similarity score (SBERT representation)
11	Prompt – answer similarity score (BOW representation)
12	Language errors

20% testing data, and train the models using 5-fold cross-validation. We evaluate the performance of these models according to the three measurement criteria defined in [24]: (i) The agreement of scores between the human raters and the computer, which has been a long-established measure of the performance of automated scoring. The agreement is represented by the quadratic weighted kappa (QWK) [8]. (ii) The degradation from human-human score agreement. This measure helps to understand whether the human-automated scoring agreement is lower or higher than the human-human (rater 1 vs. rater 2) score agreement. (iii) The standardized mean score difference between human and automated scores.

5. EXPERIMENTAL RESULTS

In this section, we analyze the individual fairness of essays to answer the problem statements mentioned in Section 2. The fairness analysis is performed by evaluating the performance of different regression models on different text features.

5.1 Evaluating Essay Similarity

5.1.1 Interpretable Essay Features

In our experiment, to understand how the distance metrics perform, we need high-level feature representations of the essays, which make them easy for humans to interpret. We examine the similarity or dissimilarity of a pair of essays using twelve interpretable essay features as implemented by [9], shown in Table 1.

Six out of twelve features of the essays are extracted using EASE (Enhanced AI Scoring Engine) library³, written by one of the winners in the ASAP Kaggle competition. This feature set has been proven to be robust [19]. EASE produces 414-length features, but we exclude the features based on BOW vectors since they are not interpretable essay features. The considered six features are answer length, word counts, average word length, the number of “good” Part of Speech (POS) n-grams, the number of overlapping tokens between the prompt and the answer, and also the number of overlapping tokens between the prompt and the answer but including the synonyms. We generate the other six interpretable essay features from the answers. Those

³<https://github.com/openedx/ease>

are the number of punctuations, the number of spelling errors, unique word count, similarity scores between answer and prompt using S-BERT and BOW vector representations, and the number of language errors.

The interpretable essay features help in providing a higher-level description of each answer. For example, language fluency and dexterity can be estimated by the average word length and long words count, according to Mahana et al. [17]. Moreover, Larkey also mentioned that the number of long words could indicate the complexity of term usage [15]. Meanwhile, unique words count is useful to estimate the richness of vocabulary in the answer, and it can exhibit students' knowledge to use different words.

We measure the relevancy of an answer against the prompt using the degree of tokens overlap between the prompt and the answer (including their synonyms), and the cosine similarity value between the answer and the prompt using the Sentence-BERT and the BOW representation. Besides, we learn the grammar feature of the text, one of which is by measuring the number of good n-grams in the essay. The EASE library extracts the answer into its POS-tags and compares them with a list of valid POS-tag combinations in English. It defines the "good" POS n-grams as the ones that separate high- from low-scoring essays, determined using the Fisher test [11]. Additionally, we count the number of language errors in each answer using Language Tool Python library⁴. Finally, we also capture the mechanics of the text, which include aspects such as the usage of punctuation and the number of spelling errors found in the answer.

5.1.2 Extreme Cases

We use the dataset that consists of 1569 essays, which means that we obtain 1,230,096 unique essay pairs. The easiest and simplest way to get the first insight about the performance of the distance metrics is by examining the comparison between the most similar essay pair and the most dissimilar essay pair. We provide the visualization of each text feature in Figure 1, which highlights the similarity (or dissimilarity) for each of the twelve interpretable essay features, using the normalized feature values. Using this graph, we can observe in which aspects the essays are either similar or dissimilar.

BOW. Firstly, we inspect the comparison of extreme cases for BOW features. Figure 1 compares the difference of feature values between the most similar and the most dissimilar pair from the BOW representation of the essays. It is evident that it cannot distinguish the nearest pair and farthest pair of essays very well based on most of the features. BOW mainly focuses on the *Prompt Similarity BOW* feature.

TF-IDF. Secondly, we investigate the comparison of extreme cases for TF-IDF features. Here, both BOW and TF-IDF have the same essay pairs for the nearest and the farthest essays. This makes sense since the TF (Term Frequency) in the TF-IDF vector is actually the BOW vector, followed by the multiplication by the inverse document frequency (IDF), and then normalized by the Euclidean norm. So, it is not surprising if both of these vectors have similar pairs.

⁴https://github.com/jxmorris12/language_tool_python

Sentence-BERT. Next, we investigate the comparison of extreme cases for Sentence-BERT features. As opposed to the BOW representation, based on our twelve interpretable features, Sentence-BERT can provide a more clear distinction between the nearest and farthest pair for most of the features. It is noticeable that for the nearest pair, both values are almost the same, contrary to the farthest pair, which shows very different values.

LASER. Finally, we investigate the comparison of extreme cases for the LASER features. Figure 1 compares the difference in feature values between the most similar and the most dissimilar pair using LASER representation of the essays. It is evident that based on our twelve interpretable features, LASER provides the best distinction between the nearest and farthest pair for most of the features. The normalized feature values for all twelve interpretable features look close to each other for the nearest pair. In contrast, the normalized feature values for the farthest pair appears very dissimilar for all of the essay features, and visibly has the largest differences compared to the other text representations.

5.1.3 Aggregate of Feature Difference

In this section, to get a wider overview of the performance of the distance metrics, we conducted the evaluation using more than just the extreme cases. For the experiment, we sampled 30,000 pairs of essays from three different distance distributions: 10,000 pairs from the nearest distance pairs, 10,000 pairs from the medium distance, and 10,000 pairs from the farthest distance. For the medium distance, we collected 5,000 pairs before the median value and the other 5,000 after the median value. In this research, the expected condition is that the average of features' differences for all pairs in the nearest distance will be lower than the average of features' differences of all pairs in the medium distance. Consequently, the average of features' differences for the medium distance should be lower than those in the farthest distance.

The results are shown in Figure 2. The x-axis is labeled using numbers from 1 to 12 that refer to the twelve interpretable essay features in Table 1, in the same order. Consistent with the results of the extreme cases discussed before, LASER outperforms the other text representation of essays. Although for feature number 12 (*Language Errors*), it is not showing the expected result. This is also the case for the Sentence-BERT vector, with only one feature having an unexpected result (feature number 7: *Number of punctuation*). All of the other features for Sentence-BERT work well, even though mostly with smaller margins than those by the LASER vector.

The performance of Bag-of-Words (BOW) and TF-IDF representations is not as good as the deep-learning-based text representations. Some of the results are counterintuitive since the average of features' differences in the nearer pairs are larger than the farther ones. However, several of the essay features worked well as the exceptions, i.e., feature numbers 3, 8, 10, 11, and 12. The complete feature names can be referred to in Table 1. It is visible that BOW and TF-IDF have similar characteristics as also the case for the extreme cases in the subsection 5.1.2.

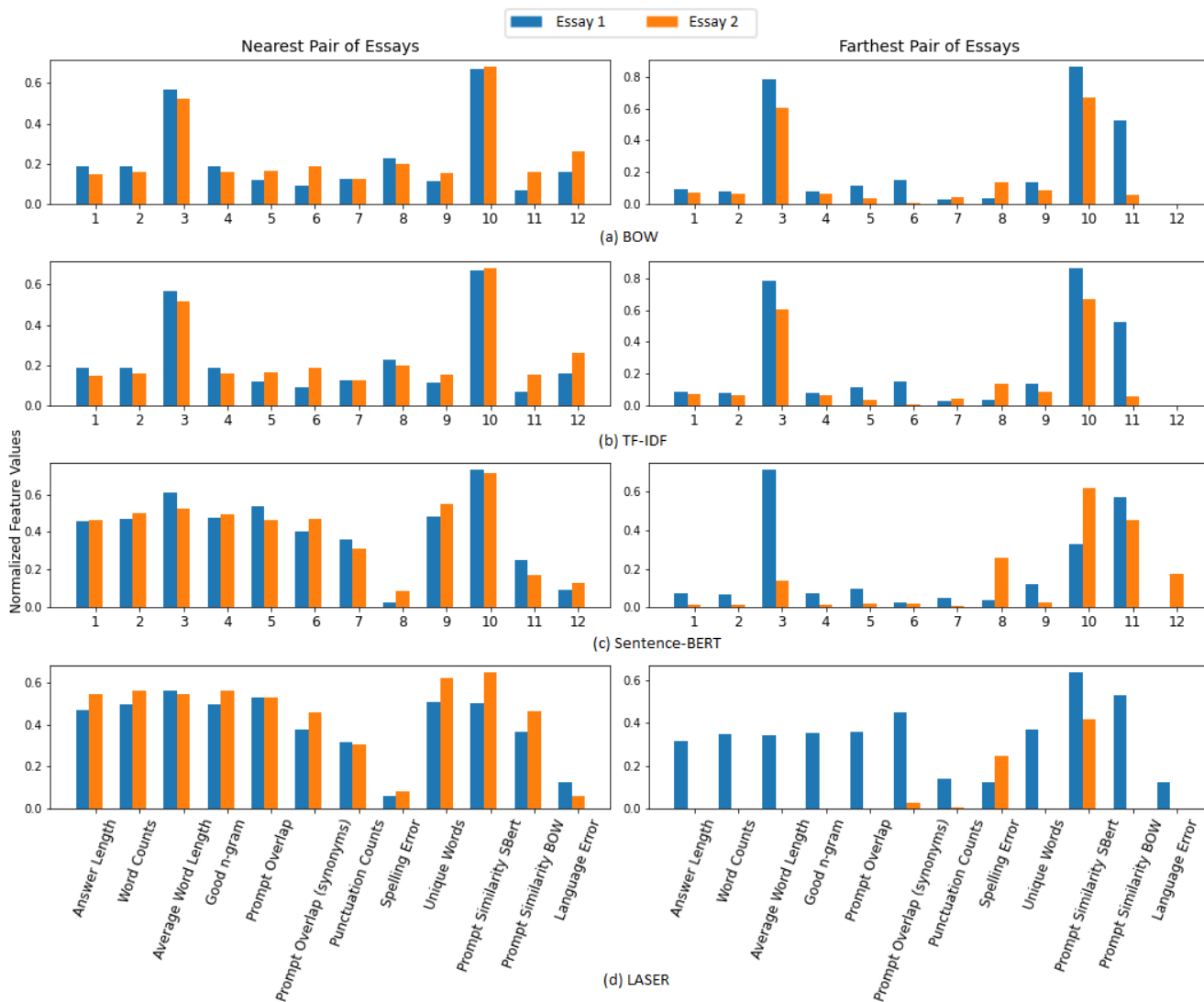


Figure 1: Nearest and Farthest Pair of Essays

5.2 Evaluation against Paraphrasing

We have evaluated the performance of the text representation with regards to their ability to measure similarities between essays. Our method proposes a further performance check concerning their ability to detect paraphrased essays. Students may write their answers using different words but still convey the same meaning. In this context, the scoring system should be able to perform consistently. We considered that two essays with the same content and arguments which were expressed using different words should be graded similarly.

We conducted this consistency check by inspecting whether the text features could maintain the similarity ranking between the original essays and their corresponding paraphrased essays. The rule is, if essay A is more similar to essay B than essay C , then the paraphrase of A also must be more similar to the paraphrase of B than the paraphrase of C .

For this purpose, the paraphrased essays were generated using the Spinbot API⁵. To understand how the API performed, a snippet of an essay along with its paraphrased version is shown below as an example:

Original essay. “One nice sunny day I was trapped in a doctors office with no air conditioning. this doctor’s office had @NUM1 chairs in the dirty waiting room. I was already cramed in the room with about @NUM2 people for @NUM3 minutes. I was trying to be patient but the patience was wearing off...”

Paraphrased essay. “One decent bright day I was caught in a specialists office with no cooling. this specialist’s office had @NUM1 seats in the grimy lounge area. I was at that point cramed in the room with about @NUM2 individuals for @NUM3 minutes. I was attempting to be patient however the tolerance was wearing off...”

During the experiment, we first sampled 100 essays each

⁵<https://api.spinbot.com>

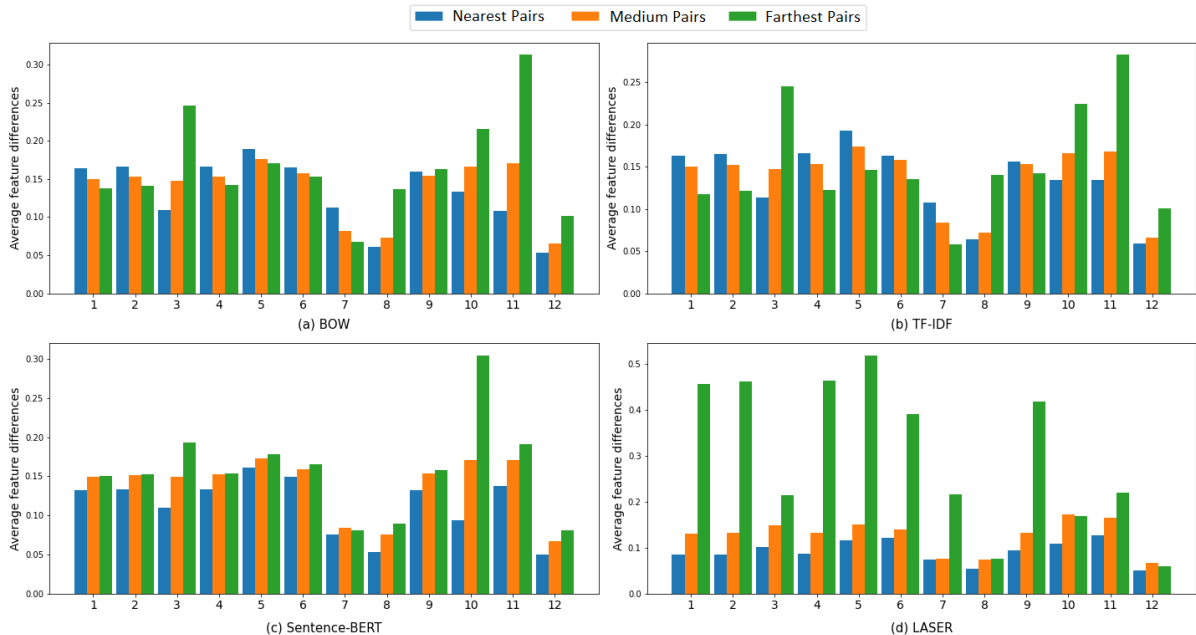


Figure 2: Aggregate Features Differences for three Distance Groups

from the original version and the paraphrased version. For each sample, we calculated the cosine similarity values between that sample with all of the other (1569) essays in the dataset. Afterward, we created a similarity ranking in descending order, from the highest similarity to the lowest. We implemented the same procedure on the paraphrased essay. Therefore, we also obtained another descending-ordered similarity ranking from the paraphrased version. From these two rankings, we measured the ability to maintain the ranking using Normalized Discounted Cumulative Gain (NDCG) score [13]. Finally, we repeated the same procedure 100 times with the other sampled essays. In the end, we obtained one hundred NDCG scores from 100 samples. Then we calculated the average NDCG score. The whole method is depicted in Figure 3.

NDCG value ranges from 0 to 1, where score 1 means a perfect ranking. In document retrievals, a score of 1 indicates that the order of document hits in a search is perfectly ordered by relevance. We calculated the average NDCG scores of 100 samples against the other essays (1569) in the dataset for each of the four text representations of the essays. The results were satisfying, as all of them had an NDCG score of 1. This means that all of the vector representations had no problem in maintaining the ranking of similarities between the original essays and the paraphrased essays.

5.3 Evaluating the Individual Fairness

5.3.1 Regression Model Results

Table 2 describes the performance for all of the regression models using different text features. Based on the vector representations of essays, it is evident that all three regression models using LASER features provide the best results. While according to the regression algorithm, Gradient Boosting performs best for all text features, followed by Ridge Regression, and lastly, Random Forest.

5.3.2 Equation-based Evaluation

Speicher et al. [22] consider individuals who belong to different sensitive groups yet share similar protected attributes should have the same decision outcomes. For instance, essays having the same protected attributes must not be treated discriminatively based on the students’ gender or race.

Individual fairness requires that the distance between two individuals’ outcomes is less than their distance according to the similarity metric. A mapping of $M : V \rightarrow \Delta(A)$ satisfies the (D, d) -Lipschitz property if for every $x, y \in V$, we have:

$$D(M(x), M(y)) \leq d(x, y) \quad (3)$$

where, V is a set of individual essays, M is a function that assigns the essays into probability distribution over the outcomes A , D is a distance function that measures the difference in the outcomes, and d represents the similarity metric between two essays.

We applied cosine distance to create the distance metric. For prompt 7 in ASAP dataset, we obtained an $N \times N$ distance metric with $N = 1569$. To have the distance metric and outcome metric in the same scale, we used Min-Max normalization to rescale the feature values into the range of $[0, 1]$. Min-Max scaler is defined as:

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}} \quad (4)$$

where x is the original value, x_{min} is the minimum value, and x_{max} is the maximum value. The cosine similarity between two vectors A and B is:

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{|\mathbf{A}| |\mathbf{B}|} = \frac{\sum_{i=1}^n \mathbf{A}_i \mathbf{B}_i}{\sqrt{\sum_{i=1}^n \mathbf{A}_i^2} \sqrt{\sum_{i=1}^n \mathbf{B}_i^2}} \quad (5)$$

For each text feature, we count the number of pairs of essays

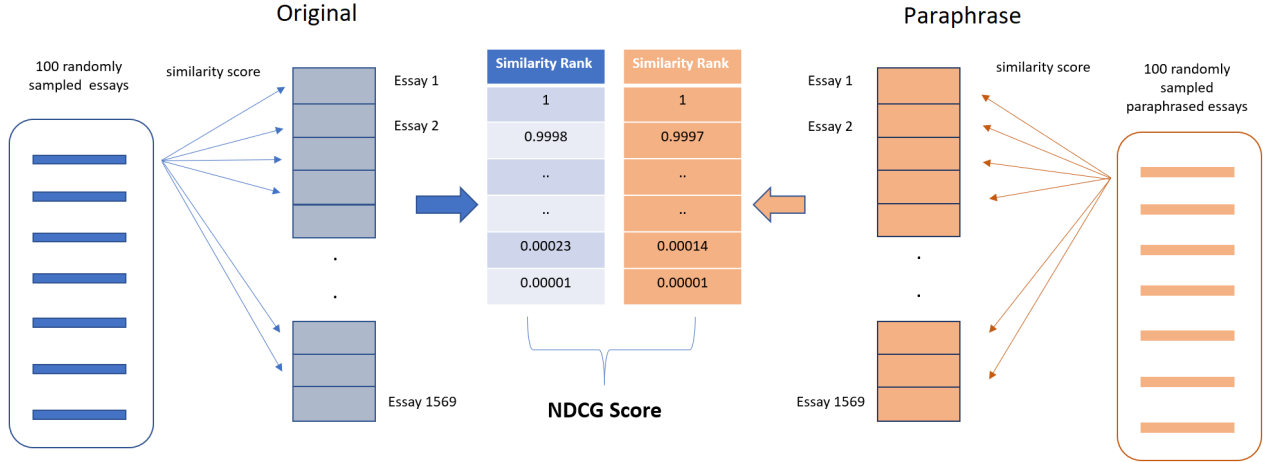


Figure 3: Ranking Evaluation from the Original and Paraphrased Essays

Table 2: Performance of Regression Models using Different Text Features

Features	Model	QWK Score	Human Agreement	Degradation	\bar{Z}
BOW	Random Forest	0.637	0.721	0.084	0.023
	Gradient Boosting	0.708	0.721	0.014	0.0063
	Ridge Regression	0.658	0.721	0.064	0.012
TF-IDF	Random Forest	0.602	0.721	0.120	0.001
	Gradient Boosting	0.703	0.721	0.018	0.004
	Ridge Regression	0.685	0.721	0.037	0.002
SBERT	Random Forest	0.618	0.721	0.104	0.036
	Gradient Boosting	0.722	0.721	-0.001	0.011
	Ridge Regression	0.699	0.721	0.022	0.028
LASER	Random Forest	0.713	0.721	0.009	0.0068
	Gradient Boosting	0.767	0.721	-0.046	0.0001
	Ridge Regression	0.758	0.721	-0.036	0.0024

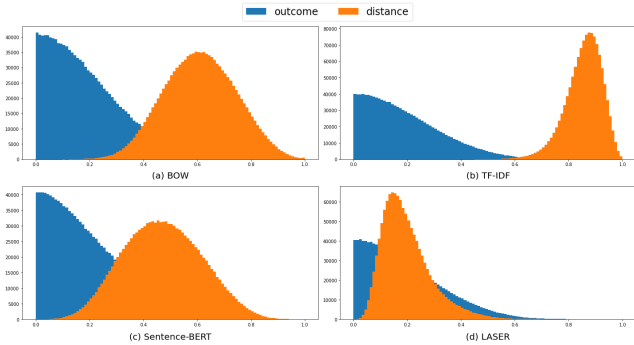


Figure 4: Score and Distance Distribution

that satisfy equation 3. The dataset contains 1,569 essays, and therefore, there are 1,230,096 unique pairs of essays. Table 3 shows the proportion of pairs of essays that satisfy the individual fairness constraint for every vector representation of essays with its corresponding regression model. It is important to note that we should read the table row by row. For example, if we applied Bag-of-Words vector as the text features, Random Forest has a higher proportion of pairs that satisfy the individual fairness equation than Gra-

dent Boosting and Ridge Regression algorithms. If we use LASER, we can see that Ridge Regression outperforms the other algorithms with regard to the equation.

It is not possible to compare the performance between different vector representations since they work on different distance distributions. We can see their distance distribution in Figure 4. The orange curve is the distance distribution, and the blue one is the outcome distribution. To have a higher proportion of pairs that satisfy the equation, a major part of the blue curve should be on the left side of the orange curve, which will indicate that the outcome distance is lower than the vector distance. It is obvious that for BOW and TF-IDF vector, the majority part of the blue curve is on the left of the orange curve, which indicates that they had more pairs of essays that satisfy the equation. Meanwhile, LASER vector distance seemed to be more distributed on the left side, thus having more pairs with lower distances than the outcomes. It helps to explain why the proportion of pairs that satisfy the equation was much lower for LASER as compared to the other text features.

5.3.3 Score Difference vs. Distance of Essays

In the presence of a large number of pairs of essays, it is impossible for humans to examine each pair one by one. We propose a method based on a simple idea by using a visual

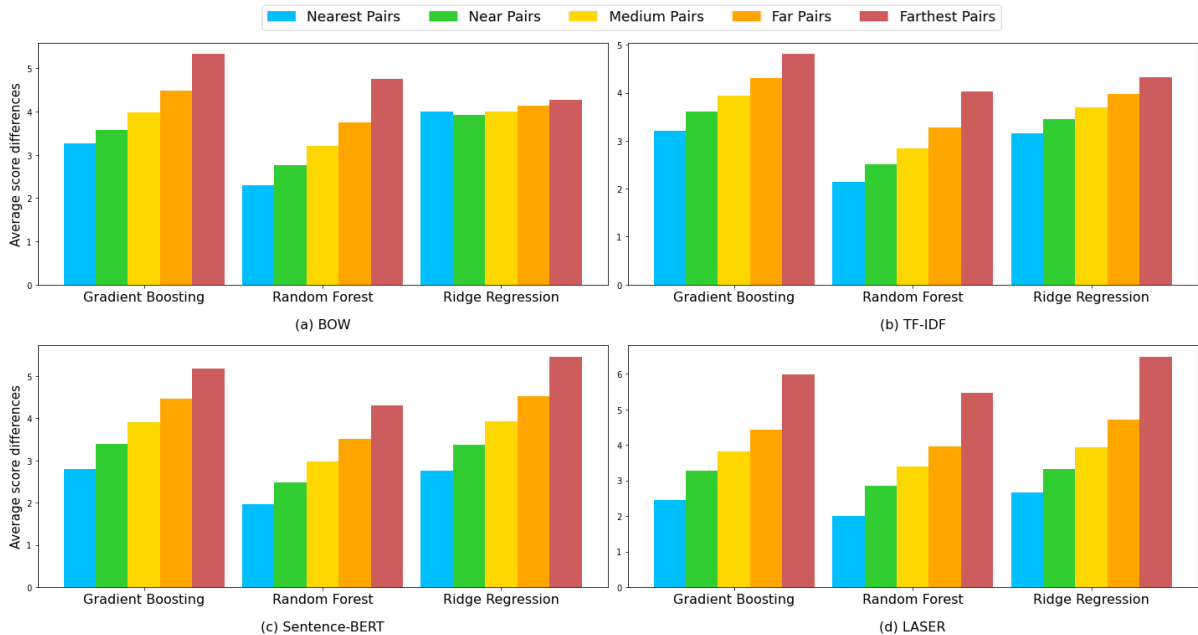


Figure 5: Average Score Differences for five Distance Groups

Table 3: The proportion of Pairs of Essays that Satisfy Equation 3 (in percentage)

Features	Random Forest	Gradient Boosting	Ridge Regression
BOW	99.4%	99%	99.2%
Tf-Idf	99.9%	99.9%	99.9%
Sentence-BERT	92.2%	94.7%	97.9%
LASER	51.7%	53.9%	67.8%

inspection of the distribution of score differences across different distance groups. Evaluating individual fairness is by looking at whether similar essays are treated similarly. It means that the score difference between essays must correlate with their distance in the vector space. The average score differences in the group of similar essays should naturally be lower than the average score difference in the group of dissimilar essays.

In our work, we presented the visualization of score difference vs. distance of essays in two scenarios, using data binning to (i) five groups (Figure 5) and (ii) one hundred groups (Figure 6). Firstly, we sorted the distance of all pairs of the essays in ascending order, from the most similar pair (the pair with the lowest cosine distance) to the most dissimilar pair (the pair having the highest cosine distance). For the data binning of five, we divided these sorted essays into five groups, with each group comprised of 20% of the whole data. For each group, first, we calculated the score differences of all pairs within that group. Afterward, we calculated the average value of those score differences. Lastly, we compared these average score differences of all five groups. We conducted this process for all of the regression models and text features of the essays. The results are shown in Figure 5. We can see that basically, all distance metrics with all three

models could produce the correct proportion of score distance and vector distance, except for the Ridge Regression algorithm on Bag-of-Words (BOW) vector, which appears flat, which means that the score difference was not correlated with the vector distance. Therefore, this does not follow the expected condition for individual fairness and is not recommended.

The exact same procedure was then repeated, but instead of dividing the whole sorted pairs into five groups, we divided them into one hundred groups to get a smoother graph. The results are shown in Figure 6. Now, we can observe a smoother increase in score differences. In Figure 5, we observe that the graph for Ridge Regression using the BOW vector appears flat. Sentence-BERT has the best shape of the increment of score differences since it grows in a gradual and consistent manner. For LASER, it shows a smooth increment at the beginning but flattens a bit towards the end and suddenly rises significantly. Moreover, if we look at the graph carefully, there is even a drop in score differences after reaching the peak by the Gradient Boosting and Random Forest algorithms. Although, this phenomenon is not seen in the Ridge Regression algorithm. TF-IDF vector exhibits a smooth increasing behavior similar to the Sentence-BERT vector. The only slight difference is that there are sharp increases at the right-most part (most dissimilar pairs). This visualization method helps quickly understand the individual fairness performance of a combination of a regression model and a text representation vector.

5.4 Discussion

We are looking for a combination of text features and regression model that has a better overall performance with respect to different aspects of our methodology. On the one hand, for the text features, they should perform well in these criteria: (i) the ability to discriminate between the most

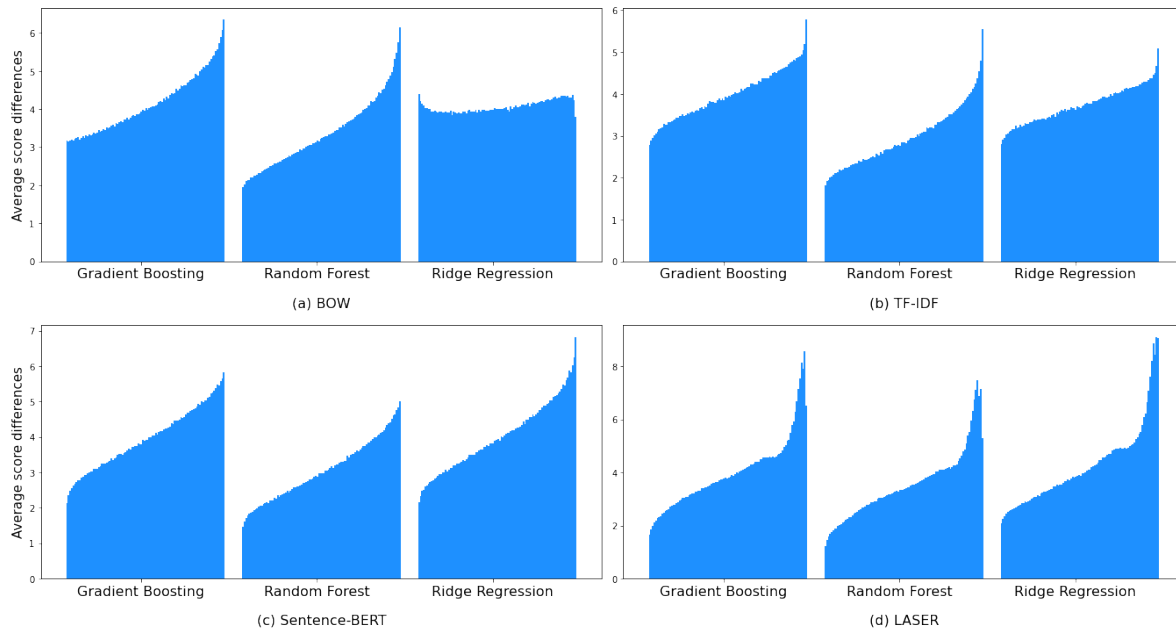


Figure 6: Average Score Differences for 100 Distance Groups (from left to right: the nearest pairs to the farthest pairs)

similar and most dissimilar pair of essays (extreme cases), (ii) the aggregate of features differences, and (iii) the ability to maintain the ranking of similarities between the original and paraphrased essays. On the other hand, the regression models should have good capabilities in these aspects: (i) the proportion of pairs that satisfy the individual fairness equation, and (ii) the correlation of score difference with the distance of essays. Ultimately, both of them must have a high agreement between the system scores and the human scores, measured in Quadratic Weighted Kappa (QWK). According to the acceptance criteria for an essay scoring model from Williamson et al. [24], the QWK must be at least 0.7.

Based on the proposed methodology, we analyze the outcomes of experimental results. LASER has the highest agreement (QWK) score compared to the other text features, as shown in Table 2. However, it performs inadequately for the proportion of pairs of essays that satisfy the individual fairness equation. Meanwhile, BOW and TF-IDF have the highest scores, as illustrated in Table 3. Nevertheless, they did not show better performances than Sentence-BERT and LASER in terms of discriminating the extreme cases and the aggregate features differences.

Finally, we observe that an automated essay scoring model trained using the Gradient Boosting algorithm on Sentence-BERT vector representation of essays has a better overall performance than the other combinations. It has exceeded the acceptance rate of the QWK score (0.722), performed well in discriminating the extreme cases and the aggregate of features differences, has a high percentage of pairs that satisfy the individual fairness equation (94.7%), and has the smoothest linear increment of score differences across the linearly divided distance groups as reflected in Figure 6. However, it is noteworthy to mention that selecting the combination depends on the priority given to a specific evaluation aspect when developing an essay scoring system.

6. CONCLUSION

In this work, we studied individual fairness in the Automated Essay Scoring (AES) system and proposed an individual fairness evaluation methodology. In practice, it is a challenging task to define the similarity of essays. We used the numerical text representation of essays to compute the similarities between answers. Next, we compared the performance of different scoring models using different text representations with regard to individual fairness. Our proposed methodology suggests that we look for the combination of the text representation of essays and score prediction models that achieves well-balanced results in several evaluation aspects. Based on the experiment results, we observed that the combination of Sentence-BERT and Gradient Boosting had overall better results than the other combinations.

The proposed methodology provides flexibility to be used with any text representations of essays and any score prediction models. Moreover, it is further expandable for a more robust evaluation process. One can assess the performance of the distance metrics not only against the paraphrased essays but also against noised textual data. For example, whether an AES system can perform well on the negation of essays is still a challenging problem in the NLP domain. It will be interesting to observe whether the essay features and distance metrics can be used to differentiate pairs of essays that have similar syntactic features but different semantics. Another interesting question for future study is how individual fairness relates to and contributes to group fairness. For instance, two students with and without a disability might provide very similar essays, but the scores need to reflect the extra effort and time of the student with the disability. In this context, it would be unfair to only take essays rather than individuals into account to measure fairness in AES.

7. REFERENCES

- [1] E. Amorim, M. Caçado, and A. Veloso. Automated essay scoring in the presence of biased ratings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 229–237, 2018.
- [2] M. Artetxe and H. Schwenk. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610, 2019.
- [3] S. Bird, E. Klein, and E. Loper. *Natural language processing with Python: analyzing text with the natural language toolkit*. ” O’Reilly Media, Inc.”, 2009.
- [4] B. Bridgeman, C. Trapani, and Y. Attali. Considering fairness and validity in evaluating automated scoring. In *The Annual Meeting of the National Council on Measurement in Education (NCME)*, 2009.
- [5] J. Burstein and M. Chodorow. Automated essay scoring for nonnative english speakers. In *Computer mediated language assessment and evaluation in natural language processing*, 1999.
- [6] D. Chandrasekaran and V. Mago. Evolution of semantic similarity—a survey. *ACM Computing Surveys (CSUR)*, 54(2):1–37, 2021.
- [7] M. Chodorow and J. Burstein. Beyond essay length: evaluating e-rater®’s performance on toefl® essays. *ETS Research Report Series*, 2004(1):i–38, 2004.
- [8] J. Cohen. Weighted kappa: nominal scale agreement provision for scaled disagreement or partial credit. *Psychological bulletin*, 70(4):213, 1968.
- [9] A. Doewes and M. Pechenizkiy. On the limitations of human-computer agreement in automated essay scoring. *International Educational Data Mining Society*, 2021.
- [10] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. Zemel. Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*, pages 214–226, 2012.
- [11] R. A. Fisher. On the interpretation of χ^2 from contingency tables, and the calculation of p. *Journal of the Royal Statistical Society*, 85(1):87–94, 1922.
- [12] F. GUO. Fairness of automates essay scoring of gmat awa. *GMAC Research Reports*, 9, 2009.
- [13] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446, 2002.
- [14] J. D. M.-W. C. Kenton and L. K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186, 2019.
- [15] L. S. Larkey. Automatic essay grading using text categorization techniques. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 90–95, 1998.
- [16] D. Litman, H. Zhang, R. Correnti, L. C. Matsumura, and E. Wang. A fairness evaluation of automated methods for scoring text evidence usage in writing. In *International Conference on Artificial Intelligence in Education*, pages 255–267. Springer, 2021.
- [17] M. Mahana, M. Johns, and A. Apte. Automated essay grading using machine learning. *Mach. Learn. Session, Stanford University*, 2012.
- [18] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [19] P. Phandi, K. M. A. Chai, and H. T. Ng. Flexible domain adaptation for automated essay scoring using correlated linear regression. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 431–439, 2015.
- [20] M. Ramamurthy and I. Krishnamurthi. Design and development of a framework for an automatic answer evaluation system based on similarity measures. *Journal of Intelligent Systems*, 26(2):243–262, 2017.
- [21] N. Reimers and I. Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019.
- [22] T. Speicher, H. Heidari, N. Grgic-Hlaca, K. P. Gummadi, A. Singla, A. Weller, and M. B. Zafar. A unified approach to quantifying algorithmic unfairness: Measuring individual & group unfairness via inequality indices. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2239–2248, 2018.
- [23] V. U. Thompson, C. Panchev, and M. Oakes. Performance evaluation of similarity measures on similar and dissimilar text retrieval. In *2015 7th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K)*, volume 1, pages 577–584. IEEE, 2015.
- [24] D. M. Williamson, X. Xi, and F. J. Breyer. A framework for evaluation and use of automated scoring. *Educational Measurement: Issues and Practice*, 31(1):2–13, 2012.
- [25] L. Ying. A research on the application of automatic essay scoring system to university’s english writing education in the era of big data: Taking pigaiwang as an example. *Studies in Literature and Language*, 10(6):84–87, 2015.
- [26] W. Zhu. A study on the application of automated essay scoring in college english writing based on pigai. In *2019 5th International conference on social science and higher education (ICSSHE 2019)*, pages 451–454, 2019.

Combining domain modelling and student modelling techniques in a single automated pipeline

Gio Picones
School of Computer Science
The University of Sydney
Sydney, Australia
gpic4558@
uni.sydney.edu.au

Irena Koprinska
School of Computer Science
The University of Sydney
Sydney, Australia
irena.koprinska@
sydney.edu.au

Benjamin Paaßen
German Research Center for
Artificial Intelligence (DFKI)
Berlin, Germany
benjamin.paassen@
dfki.de

Kalina Yacef
School of Computer Science
The University of Sydney
Sydney, Australia
kalina.yacef@
sydney.edu.au

ABSTRACT

In this paper, we propose a novel approach to combine domain modelling and student modelling techniques in a single, automated pipeline which does not require expert knowledge and can be used to predict future student performance. Domain modelling techniques map questions to concepts and student modelling techniques generate a mastery score for a concept. We conducted an evaluation using six large datasets from a Python programming course, evaluating the performance of different domain and student modelling techniques. The results showed that it is possible to develop a successful and fully automated pipeline which learns from raw data. The best results were achieved using alternating least squares on hill-climbing Q-matrices as domain modelling and exponential moving average as student modelling. This method outperformed all baselines in terms of accuracy and showed excellent run time.

Keywords

computer science education, domain modelling, student modelling, Q-matrix, mastery score, task-sequencing

1. INTRODUCTION

The idea of mastery learning was first proposed by Bloom [3] as an educational philosophy based on the belief that nearly all students can master a studied subject when given enough time and support. Nowadays, the term more commonly refers to an educational approach where test questions assess certain concepts and a mastery of prerequisite concepts is required before moving to harder concepts [11]. A con-

cept or skill (we use these terms interchangeably) can be thought of as a unit of knowledge that is assessed by a question; for example, a mathematics question might assess the concepts of "addition" and "subtraction", or a programming question might assess the concepts of "print statements" and "strings" [7]. A common application of this approach is for task sequencing in intelligent tutoring systems, where remedial questions for students are selected based on concepts they are weak at [11]. These approaches can also be used for providing automated feedback to students and hint generation [8]. But how do we know the full set of concepts in a set of questions? How do we know which questions assess which concepts? How do we measure the extent to which a student has "mastered" each concept? There are two distinct but related fields of research that attempt to answer these questions—domain modelling and student modelling [12].

Domain modelling is concerned with knowing the full set of concepts in a set of questions, and knowing which questions assess which concepts. This mapping of questions to concepts can be represented as a **Q-matrix**, a m questions \times k concepts matrix, where each entry is either 1 or 0, with 1 representing that the question in the row assesses the concept in the column [13]. Several previous studies have proposed methods for generating a Q-matrix based on student performance data on question sets; e.g. Barnes [2] introduced a hill-climbing algorithm, and Desmarais et al. [5] proposed a method in which Alternating Least Squares (ALS) method is used to refine an expert-designed Q-matrix.

Student modelling is concerned with measuring the extent to which a student has mastered a concept, or how to generate a **mastery score**. For instance, Corbett et al. [4] introduced Bayesian Knowledge Tracing, which uses a Markov model with two hidden states (a concept is either 'known' or 'unknown' by a student), with further refinements to this technique proposed by Pardos et al. [9] and Yudelson et al. [14].

In this paper, we aim to combine domain and student modelling in a single, automated pipeline. In other words, is it

G. Picones, B. Paaßen, I. Koprinska, and K. Yacef. Combining domain modelling and student modelling techniques in a single automated pipeline. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 217–227, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853131>

print	variable_assignment	if_conditions	inbuilt_functions	def_functions	collections	loops	file_io
1	1	0	1	0	0	0	0
1	1	0	1	0	0	0	0
1	1	1	1	0	0	0	0
1	1	1	1	0	0	0	0
1	1	1	1	0	0	0	0
1	1	1	1	0	0	0	0
1	1	1	1	0	0	0	0
1	1	1	1	0	1	0	0
1	1	1	1	0	1	1	0
1	1	0	1	0	1	1	0
1	1	1	1	0	0	1	0
1	1	0	1	0	0	1	0
1	1	1	1	0	1	1	0
1	1	0	1	0	1	1	0
1	1	1	1	0	1	1	0
1	1	1	1	0	1	1	0
1	1	1	1	0	1	1	0
1	1	1	1	0	1	1	1
1	1	1	1	0	1	1	1
1	1	0	1	1	0	0	0
0	1	0	1	1	0	0	0
0	1	0	0	1	1	0	0
1	1	0	1	0	1	1	0
1	1	1	1	0	1	1	0

Figure 1: Expert Q-matrix for Intermediate-2018. Questions are rows and concepts/skills are columns.

possible to go from domain modelling to student modelling in a single pipeline on a single dataset, and thus predict future student performance? Solving this problem would be of significant benefit for intelligent tutoring systems, for example for task sequencing. Using only historic student performance data on questions, it would be possible to automatically generate a mapping between questions and concepts, and generate a mastery score for each student for each concept. This can then be used to sequence remedial tasks for students, i.e., for recommending tasks which assess the concepts at which the students were the weakest, with minimal human input.

This study attempts to explore this question, and has three main contributions. First, it provides an implementation and analysis of a pipeline that includes different combinations of three domain modelling and three student modelling techniques, where the results are based on predicting future student performance using the mastery score. Second, it performs this analysis on large datasets from an online programming course - 6 datasets across 2 years with 3 levels of expertise (beginners, intermediate and advanced), with a total of 144 questions and 28,466 students - demonstrating consistent results across all datasets. Third, it introduces a new Q-matrix generation technique - ALS on Hill-Climbing Q-matrix. The results show that the proposed technique is the most successful student modeling - it is best combinations of domain and student modelling in 5/6 cases.

2. RELATED WORK

2.1 Domain Modelling (Q-matrix Generation Algorithms)

A Q-matrix is a mapping between questions in a set, and the skills or concepts that they assess [13]. An example is shown in Fig. 1. Such matrices are used to determine the probabilities that certain skills are learned by a student, and to select and sequence tasks in mastery learning environments. While Q-matrices are traditionally designed by experts, we have to consider disagreement between experts, the amount of time and effort needed to do this process manually, as well as disagreement between expert opinion and what students

actually experience.

Barnes [2] introduced an algorithm that can derive a Q-matrix from student response data. The algorithm initializes a $m \times k$ matrix (m being the number of students, k the number of skills) with random values for each cell in range $[0, 1]$. Given the current Q-matrix, it computes, for each student, the binary knowledge states which best describe the student’s actual test responses. Then, hill-climbing is performed to improve the Q matrix given the current knowledge state estimate. This is repeated until convergence, and repeated for different numbers of concepts.

By contrast, Desmarais et al. [5] propose to let an expert design the initial Q-matrix, and then optimising via the ALS method. This is motivated by expressing the $m \times n$ matrix of student responses R (n being the number of students) as a product of two matrices Q and S ; the Q matrix being an $m \times k$ matrix as before, and the S matrix being a $k \times n$ matrix that represents student mastery profiles. The ALS method minimizes the squared error $\|R - Q \cdot S\|^2$ with respect to Q and S in an alternating fashion until convergence.

In this paper, we propose a new method which combines hill-climbing and ALS, called ALS on Hill-climbing Q-Matrix. It uses a hill-climbing algorithm to derive an initial Q-matrix, and then ALS to refine it. We evaluate the performance of three methods: Hill-climbing Q-matrix, ALS on Expert Q-matrix and ALS on Hill-climbing Q-matrix.

2.2 Student Modelling (Mastery Score Algorithms)

Once a Q-matrix is known, the students’ learning process can be described by modelling which skill is mastered at what time by which student.

Kelly et al. [6] conducted a study to determine the accuracy of two methods, namely N -Consecutive Correct Responses (N -CCR) and Knowledge Tracing (KT) [4] in detecting concept mastery in students. The study defined mastery of a concept as a binary attribute. N -CCR is a simple algorithm that considers a skill mastered once a student gives N consecutive correct responses on questions related to the skill. By contrast, KT models student knowledge as a latent variable in a Hidden Markov Model, with the parameters updated based on whether or not the student gets each question in a sequence of questions correct or wrong (more detail on this below). The algorithm computes the probability that a student has mastered a skill after each observation of a correct or incorrect answer. The study concluded by stating that 3-CCR was the better approach for next-problem correctness, and 5-CCR was the better approach for predicting performance on a transfer task, with both methods being more accurate than knowledge tracing.

However, N -CCR has drawbacks which are not discussed in the paper. Consider two students who answered a sequence of 6 questions, and assume we consider 5-CCR, with the vectors representing the correctness of the answers of each student being $[0, 0, 0, 0, 0, 0]$ and $[1, 1, 1, 1, 0, 1]$. 5-CCR would determine that both students have the same level of mastery, which can be argued is not true. One way to mitigate this is to introduce a method where more recent answers

Table 1: Dataset statistics

	Questions	Students
Beginners-2018	29	7956
Intermediate-2018	25	4756
Advanced-2018	22	731
Beginners-2019	28	8662
Intermediate-2019	25	5423
Advanced-2019	15	938

carry more weight in determining the measure of mastery. The exponential moving average (EMA) technique [10] estimates the mastery of a student after answering the t th question via the expression $\sum_{\tau=0}^{\infty} r_{t-\tau} \cdot w_{\tau}$, where $r_{t-\tau}$ is 1 if the student answered question $t - \tau$ correctly and 0, otherwise, and where $w_{\tau} = (1 - \lambda) \cdot \lambda^{\tau}$ is a weight for past states, with $\lambda \in (0, 1)$ being a hyperparameter that controls how fast this weight decays into the past.

Corbett et al. [4] introduced the concept of Bayesian Knowledge Tracing (BKT). BKT assumes that a concept is a latent binary variable (either mastered or not mastered by a student) in a Hidden Markov Model, with observations also being binary (a student can get a question correct or incorrect). There are 4 parameters in this model; the probability $p(L_0)$ that a student has mastered the skill prior to attempting the first question, the probability $p(T)$ that a student will master the concept after one opportunity to apply the concept in a question, the probability $p(G)$ that a student will get a question correct given they have not mastered a concept ('guess'), and the probability $p(S)$ that a student will get a question incorrect given that they have mastered a concept ('slip'). One model is built for each concept k , assuming that the concepts assessed in a given question are known.

In this paper, we will evaluate all three approaches: N -CCR, EMA and BKT.

3. DATA AND PREPROCESSING

The data used in this study was provided by Grok Academy (groklearning.com) and comes from three online Python programming courses for high school students. In total it comes from 28,466 students and includes 144 questions. In this learning platform, students complete coding tasks and receive automated feedback in the form of passed test cases and feedback about the failed cases, e.g. comparison between the expected output and the student's output.

Our study uses a total of 6 datasets from 2 years (2018 and 2019), with each year having one set of questions from 3 difficulty levels (beginners, intermediate and advanced). Table 1 summarises the dataset statistics.

For each student, we collected a 'trace' of all their submissions on each question. A student may submit solutions multiple times for a question, each time getting feedback in terms of expected vs actual program output, and the number of test cases passed. However, domain and student modelling approaches typically require a single binary value of correctness for each student on each question. To obtain such a value, we used 1 if the average number of passed test

Table 2: Runtimes for generating 10 Q-matrices (with 1 to 10 concepts) per dataset via hill-climbing

Dataset	Hill-climbing Q-matrix Runtime
Beginners-2018	5 hours
Intermediate-2018	4 hours
Advanced-2018	30 minutes
Beginners-2019	7 hours
Intermediate-2019	5 hours
Advanced-2019	45 minutes

cases per submission exceeded the median, and 0, otherwise. For example, if a student attempts a question 3 times and passes 0/10, 5/10 then 10/10 test cases, then we take the average over the 3 attempts, which is 5/10, and map this to 1 if 5/10 is larger than the median value over all students for this question.

4. METHOD

There are two main steps, namely domain modelling and student modelling. The first step infers the relation between questions and concepts in a form of Q-matrix. The second step builds a model to predict a student's future performance based on their past responses.

We consider three algorithms for domain modelling, namely Hill-climbing Q-matrix [2], ALS on Expert Q-matrix [5], and ALS initialized with Hill-climbing Q-matrix. The last one is our proposed method and it combines hill-climbing and ALS algorithms. Further, we consider three algorithms for student modelling, namely N -CRR, EMA, and BKT. The pipeline for each combination of domain modelling and student modelling components is as follows.

First, infer a Q-matrix using the domain modelling component. Second, for each student, extract one sequence of correct and wrong responses to questions that are related to that skill. Finally, use the student modelling component to model the mastery of each student for each skill.

For prediction, we consider the mastery score after responding to the t th question and predict a successful response if the mastery score exceeds a threshold. Our evaluation measure is the RMSE between the mastery scores and the actual responses.

5. EXPERIMENTAL SETUP

Combining each domain modelling algorithm (Hill-climbing Q-matrix, ALS on Expert Q-matrix, ALS on Hill-climbing Q-matrix) with each student modelling algorithm (N -CCR, EMA, BKT) yields nine combinations. We evaluate each combination on each data set from Table 1. For brevity and lack of space reasons, we discuss in detail the 2018 datasets but provide only the best results for the 2019 datasets.

Using hill-climbing, we generate 10 Q matrices for each dataset with $k = 1$ to $k = 10$ concepts. The runtime needed for the Q matrix generation is shown in Table 2.

To evaluate the student modelling, we use five-fold cross validation over students. For N -CCR and EMA, we also

Table 3: Performance of the three baselines (RMSE) on the 2018 data sets

Baseline Method	beginner	intermediate	advanced
random	0.7065	0.7054	0.7131
majority	0.6576	0.6447	0.5270
BKT-1	0.4899	0.4880	0.4280

Table 4: Performance of BKT, EMA and N -CCR when combined with Hill-climbing Q-matrix - RMSE on Beginners-2018 dataset

n_con	BKT	EMA (λ , RMSE)	N -CCR (N , RMSE)
1	0.4453	0.1, 0.1487	1, 0.1482
2	0.4316	0.1, 0.1588	1, 0.1594
3	0.4352	0.1, 0.2476	1, 0.2484
4	0.4391	0.1, 0.2261	1, 0.2276
5	0.4368	0.1, 0.2274	1, 0.2286
6	0.4351	0.1, 0.2280	1, 0.2293
7	0.4340	0.1, 0.2193	1, 0.2208
8	0.4318	0.1, 0.2114	1, 0.2131
9	0.4305	0.1, 0.2463	1, 0.2491
10	0.4309	0.1, 0.2389	1, 0.2417
(sec)	54.1	49.7	48.5

evaluate the results for different hyperparameter values: N from 1 to 10 and λ from 0.1 to 0.9.

Additionally, we compare the performance with three baselines, namely random guessing (random), predicting the majority label for each question (majority), and BKT with a single concept per question (BKT-1).

All six techniques were implemented in Python. For hill-climbing, ALS, N -CCR, and EMA we used our own implementation, and for BKT we used the pybkt toolbox [1].

6. EXPERIMENTAL RESULTS

6.1 Beginners-2018

The performance of the three baselines on the 2018 datasets is shown in Table 3. We can see that the most accurate baseline on all 3 datasets is BKT-1, followed by majority and random.

Table 4 shows the RMSE for BKT, EMA, and N -CCR on the Beginners-2018 dataset for Q-matrices that were generated via hill-climbing; the best results are shown in bold. For EMA and N -CCR, we only report the results for the best-performing values of λ and N (the results for other hyperparameter values are shown in Table 5). Each row represents the results obtained for a certain number of concepts. Here, it is interesting to see that, for EMA and N -CCR, the best Q-matrix is the one with only 1 concept, while, for BKT, the best Q-matrix had 9 concepts. The RMSE stays relatively stable for the different Q-matrices for BKT. For BKT and N -CCR, the RMSE stays relatively stable only for Q-matrices with 3 concepts and above. The last row of the table shows the prediction runtime; we can see that all methods were fast and had similar prediction runtime - 48-54 seconds.

Table 5: Hyperparameter values of EMA (λ) and N -CCR (N) vs RMSE for n_con = 1 when combined with Hill-climbing Q-matrix on Beginners-2018 dataset

λ (EMA)	RMSE	N (N -CCR)	RMSE
0.1	0.1487	1	0.1482
0.2	0.1624	2	0.1742
0.3	0.1857	3	0.2068
0.4	0.2148	4	0.2524
0.5	0.2470	5	0.3067
0.6	0.2809	6	0.3231
0.7	0.3159	7	0.3059
0.8	0.3523	8	0.3893
0.9	0.3927	9	0.4257

Table 6: Performance of BKT, EMA and N -CCR when combined with ALS on Expert Q-matrix - RMSE on Beginners-2018 dataset

n_con	BKT	EMA (λ , RMSE)	N -CCR (N , RMSE)
8	0.3908	0.3, 0.1874	1, 0.2176
(sec)	12.3	6.78	6.24

It is also interesting to note that, regardless of the number of concepts, $\lambda = 0.1$ and $N = 1$ are the best-performing hyperparameters for EMA and N -CCR, respectively (see Table 4). In other words, it was always best to predict the performance on the next question based on the question immediately before it. Table 5 provides more details about the impact of the hyperparameters $\lambda = 0.1$ and $N = 1$ on the RMSE. We see a clear trend: as λ and N increase, RMSE rises as well.

Next, we evaluate BKT, EMA, and N -CCR on the Q-matrix derived by ALS when initialized with an expert Q-matrix. The expert Q-matrix covers eight skills (refer to Fig. 1). Table 6 shows the results, indicating that EMA with $\lambda = 0.3$ performs best. Table 7 shows the performance of EMA and N -CCR for different hyperparameter choices. For N -CCR we observe the same trend as before, for EMA we observe that low λ -values generally perform better, with a minimum at 0.3.

Finally, we evaluate BKT, EMA, and N -CCR on the Q-

Table 7: Hyperparameter values of EMA (λ) and N -CCR (N) vs RMSE for n_con = 8 when combined with ALS on Expert Q-Matrix on Beginners-2018 dataset

λ (EMA)	RMSE	N (N -CCR)	RMSE
0.3	0.1874	1	0.2176
0.4	0.1874	2	0.2468
0.2	0.1905	3	0.2725
0.5	0.1909	4	0.2908
0.1	0.1966	5	0.3086
0.6	0.1986	6	0.3254
0.7	0.2122	7	0.3373
0.8	0.2340	8	0.3497
0.9	0.2676	9	0.3553

Table 8: Performance of BKT, EMA and N -CCR RMSE when combined with ALS on Hill-climbing Q-matrix on Beginners-2018 dataset

n_con	BKT	EMA (λ , RMSE)	N -CCR (N , RMSE)
1	0.4262	0.3, 0.1319	1, 0.1482
2	0.4054	0.5, 0.2164	1, 0.2507
3	0.4196	0.4, 0.1937	1, 0.2220
4	0.4260	0.3, 0.1760	1, 0.1901
5	0.4210	0.3, 0.1963	1, 0.2103
6	0.4232	0.2, 0.2094	1, 0.2196
7	0.4199	0.3, 0.2216	1, 0.2331
8	0.4231	0.3, 0.1714	1, 0.1861
9	0.4215	0.4, 0.1829	1, 0.2017
10	0.4214	0.4, 0.1945	1, 0.2129
(sec)	53.2	55.8	52.1

Table 9: Hyperparameter values of EMA (λ) and N -CCR (N) vs RMSE for n_con = 8 when combined with ALS on Hill-climbing Q-matrix on Beginners-2018 dataset

λ (EMA)	RMSE	N (N -CCR)	RMSE
0.3	0.1319	1	0.1482
0.4	0.1327	2	0.1542
0.2	0.1350	3	0.1580
0.5	0.1396	4	0.1627
0.1	0.1407	5	0.1641
0.6	0.1556	6	0.1898
0.7	0.1839	7	0.1953
0.8	0.2289	8	0.2106
0.9	0.2971	9	0.2348

matrix derived by ALS when initialized with a hill-climbing Q-matrix. Table 8 shows the results. Again, EMA with $\lambda = 0.3$ performs best. Indeed, the results for EMA with $\lambda = 0.3$ are better than any other result we obtained across all domain modelling approaches. Interestingly, different λ values are optimal, depending on the number of concepts.

In Table 9, we see that the RMSE of N -CCR steadily worsens as N increases. By contrast, the RMSE for EMA has a minimum at $\lambda = 0.3$ and increases for smaller or larger values.

6.2 Intermediate-2018

From Table 10, it is interesting to note that the best models use Q-matrices with more concepts than seen in the previous dataset. For BKT, 3 concepts is optimal, for EMA and N -CCR it is 10 concepts. We also observe that, for EMA and N -CCR, the more concepts we consider, the better performance gets (considering only the model with the best parameters), which is something that we did not observe in the previous dataset. While N -CCR still selects 1 as the best N for all the Q-matrices, for EMA, most of the Q-matrices give either 0.3 or 0.2 as the best λ .

From Table 11, we can see that λ s of 0.1, 0.2 and 0.3 actually have very similar RMSE, with 0.4 and above steadily worsening in RMSE as λ increases. N -CCR exhibits the pattern of RMSE worsening as N increases.

Table 10: Performance of BKT, EMA and N -CCR when combined with Hill-climbing Q-matrix - RMSE on Intermediate-2018 dataset

n_con	BKT	EMA (λ , RMSE)	N -CCR (N , RMSE)
1	0.4364	0.3, 0.3793	1, 0.3925
2	0.4037	0.3, 0.2977	1, 0.3083
3	0.4012	0.3, 0.2751	1, 0.2842
4	0.4050	0.3, 0.2610	1, 0.2726
5	0.4021	0.3, 0.2469	1, 0.2578
6	0.4045	0.3, 0.2433	1, 0.2541
7	0.4072	0.3, 0.2370	1, 0.2455
8	0.4068	0.3, 0.2306	1, 0.2389
9	0.4076	0.2, 0.2213	1, 0.2281
10	0.4060	0.2, 0.2175	1, 0.2242
(sec)	29.1	28.7	28

Table 11: Hyperparameter values of EMA (λ) and N -CCR (N) vs RMSE for n_con = 10 when combined with Hill-climbing Q-matrix on Intermediate-2018 dataset

λ (EMA)	RMSE	N (N -CCR)	RMSE
0.2	0.2175	1	0.2242
0.3	0.2176	2	0.2354
0.1	0.2197	3	0.2429
0.4	0.2203	4	0.2573
0.5	0.2265	5	0.2758
0.6	0.2373	6	0.2989
0.7	0.2543	7	0.3217
0.8	0.2793	8	0.3550
0.9	0.3127	9	0.3870

Table 12: Per-concept RMSE for BKT, EMA and N -CCR (best models) for Hill-climbing Q-matrix on intermediate 2018 dataset

Skill	RMSE (BKT)	Skill	RMSE (EMA)	Skill	RMSE (N -CCR)
0	0.4366	0	0.3803	0	0.3925
1	0.3851	1	0.1844	1	0.1897
2	0.3970	2	0.2245	2	0.2282
		3	0.2155	3	0.2339
		4	0.1790	4	0.1854
		5	0.2279	5	0.2339
		6	0.1856	6	0.1854
		7	0.1792	7	0.1854
		8	0.1137	8	0.1071
		9	0.1786	9	0.1854

Table 13: Performance of BKT, EMA and N -CCR when combined with ALS on Expert Q-matrix - RMSE on Intermediate-2018 dataset

n_con	BKT	EMA (λ , RMSE)	N -CCR (N , RMSE)
8	0.3930	0.4, 0.2815	1, 0.1977
(sec)	8.01	4.95	4.33

Table 14: Hyperparameter values of EMA (λ) and N -CCR (N) vs RMSE for n_con = 8 when combined with ALS on Expert Q-matrix on Intermediate-2018 dataset

λ (EMA)	RMSE	N (N -CCR)	RMSE
0.4	0.2815	1	0.1977
0.5	0.2821	2	0.2009
0.3	0.2821	3	0.2093
0.2	0.2837	4	0.2145
0.6	0.2844	5	0.2205
0.1	0.2861	6	0.2247
0.7	0.2899	7	0.2365
0.8	0.3018	8	0.2481
0.9	0.3265	9	0.2636

From Table 12, we see that, across all algorithms, the per-skill RMSE is not uniform at all. For example, for N -CCR, skill 8 has an RMSE of 0.1071, while skill 0 has an RMSE of 0.3925. This can be interpreted as being caused by a large variance in student proficiency for certain skills, or that some skills are a lot harder to learn than others.

From Table 13, we see that N -CCR performs better than EMA. N -CCR has chosen 1 as the best N , while EMA has 0.4 as the best λ .

From Table 14, we see that—while there is no pattern of worsening RMSE as λ increases for EMA—the first 7 values of λ have very very similar RMSE (differences lesser than 0.01). For N -CCR we see that RMSE steadily worsens as N increases. It is also interesting to note that the worst N -CCR model (RMSE = 0.2636) performs better than the best EMA model (RMSE = 0.2815).

From Table 15, we see again that per-concept RMSE has high variance, more so with EMA and N -CCR than BKT. Skill 7 has the highest RMSE when EMA is used, with an RMSE of 0.5888. Meanwhile, the skill with the worst RMSE

Table 15: Per-concept RMSE for BKT, EMA and N -CCR (best models) on Intermediate-2018 dataset

Skill	RMSE (BKT)	Skill	RMSE (EMA)	Skill	RMSE (N -CCR)
0	0.4071	0	0.2691	0	0.1854
1	0.4105	1	0.1787	1	0.1854
2	0.4040	2	0.2079	2	0.2748
3	0.4061	3	0.1787	3	0.1854
4	0.3618	4	0.1788	4	0.1071
5	0.3776	5	0.1787	5	0.1854
6	0.3246	6	0.2079	6	0.1854
7	0.3374	7	0.5888	7	0.2280

Table 16: Performance of BKT, EMA and N -CCR when combined with ALS on Hill-climbing Q-matrix - RMSE on Intermediate-2018 dataset

n_con	BKT (RMSE)	EMA (λ , RMSE)	N -CCR (N , RMSE)
1	0.4122	0.3, 0.1782	1, 0.1854
2	0.3955	0.4, 0.2341	1, 0.2549
3	0.4152	0.4, 0.2215	1, 0.2377
4	0.4117	0.6, 0.2608	1, 0.2952
5	0.4097	0.6, 0.2448	1, 0.2803
6	0.3990	0.5, 0.2453	1, 0.2693
7	0.3980	0.5, 0.2425	1, 0.2636
8	0.3995	0.5, 0.2593	1, 0.2865
9	0.3998	0.4, 0.2289	1, 0.2465
10	0.4050	0.4, 0.2397	1, 0.2621
(sec)	27.2	31.8	30

Table 17: Hyperparameter values of EMA (λ) and N -CCR (N) vs RMSE for n_con = 1 when combined with ALS on Hill-climbing Q-matrix on Intermediate-2018 dataset

λ (EMA)	RMSE	N (N -CCR)	RMSE
0.3	0.1782	1	0.1854
0.4	0.1787	2	0.1877
0.2	0.1790	3	0.1889
0.5	0.1801	4	0.1896
0.1	0.1813	5	0.2198
0.6	0.1830	6	0.2220
0.7	0.1900	7	0.2248
0.8	0.2092	8	0.2289
0.9	0.2582	9	0.2335

for N -CCR is skill 2 which only has an RMSE of 0.2748.

It is interesting that, compared to the hill-climb Q-matrix without ALS refinement, the results in Table 16 show that ALS on Hill-climbing Q-matrix perform best when using Q-matrices with a smaller numbers of concepts.

From Table 17, we see the familiar pattern of N -CCR selecting 1 to be the best N regardless of number of concepts, while EMA has many different values of λ being the best for different numbers of concepts. The largest λ found is 0.6 when we use the Q-matrix with 5 concepts - this yields an RMSE of 0.2608, which in fact is better than even the best BKT model by a decent margin.

6.3 Advanced-2018

From Table 18, it is interesting to see that a large number of concepts (6) was best for EMA and N -CCR as compared to the Beginner and Intermediate datasets. BKT, on the other hand, performed best with only 2 concepts. The optimal λ and N were at 0.1 and 1, respectively.

Another interesting finding is that a different N is optimal for N -CCR, depending on the number of concepts. For 1 or 2 concepts, the optimal N is 2 instead of 1.

From Table 19, we do see the pattern for both EMA and N -CCR where performance worsens as the respective parameters increase in value, with RMSE increasing rapidly

Table 18: Performance of BKT, EMA and N -CCR when combined with Hill-climbing Q-matrix - RMSE on Advanced-2018 dataset

n_con	BKT	EMA (λ , RMSE)	N -CCR (N , RMSE)
1	0.3573	0.1, 0.2117	2, 0.2115
2	0.3439	0.2, 0.2130	2, 0.2178
3	0.3580	0.2, 0.2215	1, 0.2254
4	0.3618	0.1, 0.2074	1, 0.2102
5	0.3557	0.1, 0.2086	1, 0.2114
6	0.3555	0.1, 0.2048	1, 0.2081
7	0.3581	0.2, 0.2308	1, 0.2346
8	0.3631	0.2, 0.2228	1, 0.2274
9	0.3636	0.2, 0.2191	1, 0.2233
10	0.3660	0.2, 0.2157	1, 0.2204
(sec)	6.22	5.55	5.33

Table 19: Hyperparameter values of EMA (λ) and N -CCR (N) vs RMSE for n_con = 6 when combined with Hill-climbing Q matrix on Advanced-2018 dataset

λ (EMA)	RMSE	N (N -CCR)	RMSE
0.1	0.2048	1	0.2081
0.2	0.2050	2	0.2168
0.3	0.2079	3	0.2191
0.4	0.2133	4	0.2215
0.5	0.2216	5	0.2241
0.6	0.2333	7	0.2295
0.7	0.2497	9	0.2305
0.8	0.2716	6	0.2309
0.9	0.2994	8	0.2310

Table 20: Per-concept RMSE for BKT, EMA and N -CCR (best models) on Advanced-2018 dataset

Skill	RMSE (BKT)	Skill	RMSE (EMA)	Skill	RMSE (N -CCR)
0	0.3589	0	0.2117	0	0.2159
1	0.3283	1	0.2127	1	0.2159
		2	0.2347	2	0.2378
		3	0.1553	3	0.1551
		4	0.2119	4	0.2159
		5	0.1787	5	0.1843

Table 21: Performance of BKT, EMA and N -CCR RMSE when combined with ALS on Expert Q-matrix - RMSE on Advanced-2018 dataset

n_con	BKT	EMA (λ , RMSE)	N -CCR (N , RMSE)
9	0.3667	0.4, 0.3116	1, 0.4160
(sec)	1.67	0.841	0.846

Table 22: Hyperparameter values of EMA (λ) and N -CCR (N) for n_con = 9 when combined with ALS on Expert Q-matrix on Advanced-2018 dataset

λ (EMA)	RMSE	N (N -CCR)	RMSE
0.4	0.3116	1	0.4160
0.3	0.3119	2	0.4173
0.5	0.3122	3	0.4186
0.2	0.3129	6	0.4274
0.6	0.3143	7	0.4274
0.1	0.3145	4	0.4286
0.7	0.3190	5	0.4286
0.8	0.3283	8	0.4323
0.9	0.3447	9	0.4327

as λ rises to 0.8 and 0.9.

From Table 20, we see that per-concept RMSE is relatively stable for EMA and N -CCR as compared to the previous dataset, with BKT in fact having very close RMSE between the two concepts. This could be interpreted as concepts having less variance in difficulty, though we have to keep in mind that this is the Advanced dataset, which has the least number of students by far (731) compared to the Beginner and Intermediate datasets of the same year (7956 and 4756). It also has the least number of questions (22) compared to Beginner and Intermediate which have 29 and 25 respectively.

From Table 21, we can see that performance across all 3 algorithms with 9 concepts from an expert Q-matrix passed through ALS is worse compared to the previous Hill-climbing Q-matrix method. N -CCR with an N of 1 performed worse than EMA with an λ of 0.4.

From Table 22, we see that there is no pattern of worsening performance as parameters increase for both EMA and N -

Table 23: Per-concept RMSE for BKT, EMA and N -CCR (best models) on Advanced-2018 dataset

Skill	RMSE (BKT)	Skill	RMSE (EMA)	Skill	RMSE (N -CCR)
0	0.3483	0	0.3251	0	0.1843
1	0.3263	1	0.1719	1	0.1843
2	0.3730	2	0.3250	2	0.1843
3	0.3675	3	0.1726	3	0.1843
4	0.3874	4	0.1727	4	0.1843
5	0.4161	5	0.1748	5	0.1843
6	0.3486	6	0.1726	6	0.1843
7	0.3352	7	0.5223	7	0.7709
8	0.3021	8	0.4796	8	0.8846

Table 24: Performance of BKT, EMA and N -CCR when combined with ALS on Hill-climbing Q-matrix - RMSE on Advanced-2018 dataset

n_{con}	BKT	EMA (λ , RMSE)	N -CCR (N , RMSE)
1	0.3793	0.7, 0.2511	2, 0.2551
2	0.3500	0.5, 0.2192	2, 0.2428
3	0.3516	0.3, 0.2556	1, 0.2655
4	0.3496	0.3, 0.1899	1, 0.1988
5	0.3480	0.5, 0.2162	2, 0.2266
6	0.3544	0.5, 0.2234	3, 0.2341
7	0.3570	0.4, 0.2135	2, 0.2290
8	0.3677	0.4, 0.2250	1, 0.2331
9	0.3641	0.2, 0.2367	1, 0.2433
10	0.3682	0.5, 0.2552	3, 0.2625
(sec)	5.69	6.09	5.88

Table 25: Hyperparameter values of EMA (λ) and N -CCR (N) for $n_{con} = 4$ when combined with ALS on Hill-climbing Q-matrix on Advanced-2018 dataset

λ (EMA)	RMSE	N (N -CCR)	RMSE
0.3	0.1899	1	0.1988
0.2	0.1900	2	0.2051
0.4	0.1927	3	0.2074
0.1	0.1930	4	0.2279
0.5	0.1985	6	0.2287
0.6	0.2074	7	0.2287
0.7	0.2200	5	0.2302
0.8	0.2380	8	0.2372
0.9	0.2647	9	0.2372

CCR. While both methods don't show much of a difference between their best and worst parameter values (within 0.03 difference for both methods), it is worth noting that the worst EMA model still outperforms the best N -CCR model.

From Table 23, we see much more variance in per-concept RMSE as compared to the previous method, though we do have to consider that this method simply also has more concepts. An extreme example of this can be seen in the table for N -CCR, in which skills 0 to 6 all have the same low RMSE of 0.1843, while skill 7 and skill 8 have RMSE values of 0.7709 and 0.8846 respectively. It could simply be that 9 concepts is too many for this dataset, even though the expert deems 9 to be the correct number, with ALS not zeroing-out any concepts.

From Table 24, we see that applying ALS on the Hill-climbing

Table 26: Per-concept RMSE for BKT, EMA and N -CCR (best models) on Advanced-2018 dataset

Skill	RMSE (BKT)	Skill	RMSE (EMA)	Skill	RMSE (N -CCR)
0	0.3346	0	0.2235	0	0.2248
1	0.3091	1	0.1723	1	0.1843
2	0.3565	2	0.1723	2	0.1843
3	0.3801	3	0.1723	3	0.1843
4	0.3352				

Q-matrices (see Table 20) reduced the number of concepts that was optimal, from 6 to 4, and in fact also improved the RMSE, for EMA and N -CCR. For BKT this increased the number of optimal concepts from 2 to 5.

It is also interesting to see that for the Q-matrix with 1 concept, the best λ for EMA is 0.7, which is the largest we have observed so far. Across the other Q-matrices the optimal λ also varies.

In addition, Table 24 shows that N -CCR has a large variance in the optimal values for N across the different numbers of concepts as compared to previous results.

From Table 25, we do see that RMSE worsens as N increases for N -CCR. Meanwhile, for EMA, RMSE stays relatively similar for the first 6 values of λ , then worsens quicker beyond 0.6.

Table 26 shows us that RMSE stays very stable across the different concepts across all 3 algorithms - very different from when we had 9 concepts (see table 25). This shows us that the optimal number of concepts is in fact small for this dataset, much smaller than the number of concepts designed by an expert.

6.4 2018 Summary

For all three datasets, the best models are EMA together with the novel Q-matrix method of applying ALS to the Hill-climbing Q-matrices (refer to Table 27). This combination does not need any expert opinion or human input, and can be fully automated, provided that example student responses are given. Also note that the runtime is relatively quick.

6.5 2019 Summary

For the 2019 datasets, 2 out of 3 datasets have ALS+Hill-climb as the best Q-matrix method, and 2 out of 3 datasets have EMA as the best Mastery Score method. All 3 combinations of Q-matrix and Mastery Score methods can be automated and require no human input.

In fact, the only Q-matrix method which requires human input is the ALS+Expert method which, of course, has the first step of initialising a Q-matrix by getting an expert to manually create one from inspecting the data/questions.

7. OVERALL RESULTS DISCUSSION

7.1 Automation

Recall that we have 3 Q-matrix generation methods - Hill-climbing, ALS on an expert-initialized Q-matrix, and ALS on a hill-climbing-initialized Q-matrix (novel combination of the first two methods). The first and third methods do not need human input and need only response data, while the second method does need human input as the initial Q-matrix is initialised by an expert.

One of the goals of this study was to investigate methods of automating the whole pipeline from data to domain modelling to student modelling to prediction (which can then be used for task sequencing), which means that it would be

Table 27: Summary of 2018 results - best models and baselines

	Best Baseline RMSE	Best Method Q-matrix : Mastery Score : RMSE : n_concepts : runtime (sec)
Beginners-2018	0.4899	ALS+Hill-climb : EMA : 0.1319 : 1 : 55.8
Intermediate-2018	0.4880	ALS+Hill-climb : EMA : 0.1782 : 1 : 31.8
Advanced-2018	0.4280	ALS+Hill-climb : EMA : 0.1899 : 4 : 6.09

Table 28: Summary of 2019 results - best models and baselines

	Best Baseline RMSE	Best Method Q-matrix : Mastery Score : RMSE : n_concepts : runtime (sec)
Beginners-2019	0.4629	ALS+Hill-climb : EMA : 0.1243 : 1 : 59.8
Intermediate-2019	0.4832	Hill-climb : EMA : 0.1130 : 1 : 37.1
Advanced-2019	0.4427	ALS+Hill-climb : N-CCR : 0.0790 : 1 : 6.4

best to require as little human input as possible at any point in the pipeline.

With this in mind, it is encouraging that, out of the 6 datasets, the best performing combinations of Q-matrix and Mastery Score algorithms all use either ALS on Hill-climbing Q-matrix or just a Hill-climbing Q-matrix, both of which do not need human input. All 3 Mastery Score algorithms do not need human input.

In fact, 5 out of 6 combinations use ALS on Hill-climbing Q-matrix which is our proposed novel method. As we have seen in the previous results, in most datasets, applying ALS on the Hill-climbing Q-matrices will bring about an improvement in RMSE on most Mastery Score algorithms, as compared to using Hill-climbing Q-matrices without ALS.

7.2 Runtime

One drawback of ALS on Hill-climbing Q-matrix is that it first requires calculating the Hill-climbing Q-matrices, which, based on our setup with multiprocessing on 10 cores, would take a few hours on the largest of our datasets (see Table 2). However, we must consider that this can be run offline and only needs to be run once.

In a real life scenario, we start with some historical data to generate the Q-matrices. Hill-climbing would take a few hours, and then applying ALS on the resulting Q-matrices would only take a few seconds.

Once this is done, then this Q-matrix can be used in conjunction with EMA or N-CCR (which perform better than BKT in our experiments) on any new student responses to predict their mastery of each concept. This only takes a few seconds. Predictions on mastery of concepts can then be used for things such as task sequencing - for example, if we identify that a student is weak at certain concepts, then we can recommend other questions which test that same concept as remedial questions.

The Q-matrix can then be updated once sufficient new data has been accumulated by running the algorithm from scratch with the new data. Again, the long runtime doesn't really matter since this can be run offline, and until it is done

running, the old Q-matrix can be used.

7.3 BKT Performance

It is interesting that, for all the datasets, BKT performed worse than EMA and N-CCR. One explanation for this could be the variance in student skill. One limitation of BKT is that, when the parameters are fit on the data, what is really being captured are patterns in the average difficulty of the questions in the dataset. As such, it is best used when students are of a similar skill, whereas student skill in our data set varies wildly, even within one difficulty level (Beginner, Intermediate, Advanced). EMA and N-CCR are much simpler algorithms, and perhaps the issues that affect BKT do not affect these two algorithms quite as much.

8. LIMITATIONS AND FUTURE WORK

8.1 Q-matrix Interpretability

One limitation of the best Q-matrix generation algorithm (ALS on Hill-climbing Q-matrix) is interpretability. With a Q-matrix that is designed by an expert, obviously it would be clear what each concept actually represents - e.g. the labels can be "for-loops", "while-loops" etc.

However, when the Q-matrix is learned from data without human intervention (as in the case of ALS on Hill-climbing Q-matrix), each concept only has an index as a label. Interpreting what each concept represents would require an expert to inspect the Q-matrix together with the questions and responses.

Nevertheless, it can be argued that this is beyond the scope of this study, since we are concerned only with whether the process can be automated, and if the pure-automation approach performs better than the expert-initialised approach (any combination with ALS on expert Q-matrix) with respect to predicting future task performance.

8.2 Alternative Preprocessing Methods

Our algorithms require the student response data to be in the form of one response per question, with the response being either correct or wrong. This is not really the case for our data, where a programming question allows multiple attempts not a single one, and correctness is measured

based on how many test cases are passed. To deal with this, we aggregated the student's attempts into a single score as explained in the preprocessing section. Other methods for preprocessing and aggregation of the student's score can be explored in future work and may result in better performance.

8.3 User Testing

An interesting direction for future work would be user testing. In this study, we measured the performance of a model by its ability to predict future task performance for a question on the same concept using the mastery score, with the hope that this could then be used for task-sequencing. It would be interesting to test this method in a real intelligent tutoring system for sequencing remedial tasks and see the proportion of students who agreed that the questions that were recommended for remedial were targeting concepts that they felt weak at. Or in other words, to test if the students agreed that the system recommended the right questions, which was one of the evaluation methods used in [2].

9. CONCLUSION

This study investigated if it is possible to combine methods from domain modelling and student modelling in a single automated pipeline that could go from raw response data on a set of questions, to learning the mapping from concepts to questions, and predicting future task performance for a student, given the student's responses to questions for a given concept.

We experimented with combinations of three Q-matrix generation methods (domain modelling) and three Mastery Score methods (student modelling), and performed experiments on 6 large datasets containing data from students learning to program in Python for 2 years (2018 and 2019) at three different levels (beginner, intermediate and advanced).

We found that for 5 out of the 6 datasets, the best combination of domain modelling and student modelling used our proposed method, ALS on Hill-climbing Q-matrix, as the domain modelling algorithm. For the last dataset (Intermediate-2019), the best method was Hill-climbing Q-matrix. For 5 out of the 6 datasets, the student modelling algorithm in the best combination was EMA, while for the last dataset (Advanced-2019) N-CCR was the best method.

Since none of the best combinations used ALS on Expert Q-matrix as the best domain modelling technique, we conclude that it is possible to fully automate the pipeline from raw data to task sequencing with no human input, relying only on learning from data. The results are promising, in terms of both prediction accuracy and runtime, and are consistent across the datasets from the two different years and three levels.

10. ACKNOWLEDGEMENTS

We would like to thank Grok Academy for providing the datasets used in this study.

References

[1] Anirudhan Badrinath, Frédéric Wang, and Zachary A. Pardos. pyBKT: An accessible python library of

Bayesian knowledge tracing models. In Francois Bouchet, Jill-Jënn Vie, Sharon Hsiao, and Sherry Sahebi, editors, *Proceedings of the 14th International Conference on Educational Data Mining (EDM)*, pages 468–474, 2021.

- [2] Tiffany Barnes. Novel derivation and application of skill matrices: The Q-matrix method. In Cristobal Romero, Sebastian Ventura, Mykola Pechenizkiy, and Ryan Baker, editors, *Handbook on educational data mining*, pages 159–172. CRC Press, Boca Raton, FL, USA, 2010.
- [3] Benjamin S Bloom. Learning for mastery. instruction and curriculum. regional education laboratory for the Carolinas and Virginia, topical papers and reprints, number 1. *Evaluation Comment*, 1(2):1–12, 1968.
- [4] Albert T. Corbett and John R. Anderson. Knowledge tracing: Modelling the acquisition of procedural knowledge. *User Modelling and User-Adapted Interaction*, 4(4):253–278, 1995.
- [5] Michel C. Desmarais and Rhouma Naceur. A matrix factorization method for mapping items to skills and for enhancing expert-based Q-matrices. In H. Chad Lane, Kalina Yacef, Jack Mostow, and Philip Pavlik, editors, *Proceedings of the 16th International Conference on Artificial Intelligence in Education (AIED)*, pages 441–450, Berlin/Heidelberg, 2013. Springer.
- [6] Kim M. Kelly, Yan Wang, Tamisha Thompson, and Neil T. Heffernan. Defining mastery: Knowledge tracing versus n-consecutive correct responses. In Olga C. Santos, Jesus Boticario, Cristobal Romero, Mykola Pechenizkiy, Agathe Merceron, Piotr Mitros, José María Luna, Marian Cristian Mihaescu, Pablo Moreno, Arnon Hershkovitz, Sebastián Ventura, and Michel Desmarais, editors, *Proceedings of the 8th International Conference on Educational Data Mining (EDM)*, pages 630–631, 2015.
- [7] Kenneth R. Koedinger, Albert T. Corbett, and Charles Perfetti. The knowledge-learning-instruction framework: Bridging the science-practice chasm to enhance robust student learning. *Cognitive Science*, 36(5):757–798, 2012.
- [8] Jessica McBroom, Irena Koprinska, and Kalina Yacef. A survey of automated programming hint generation: The HINTS framework. *ACM Computing Surveys*, 54(8):1–27, 2021.
- [9] Zachary A. Pardos and Neil T. Heffernan. Modeling individualization in a Bayesian networks implementation of knowledge tracing. In Paul De Bra, Alfred Kobsa, and David Chin, editors, *Proceedings of the 18th International Conference on User Modeling, Adaptation, and Personalization (UMAP)*, pages 255–266, Berlin/Heidelberg, 2010. Springer.
- [10] Radek Pelánek. Application of time decay functions and the elo system in student modeling. In John Stamper, Zachary Pardos, Manolis Mavrikis, and Bruce McLaren, editors, *Proceedings of the 7th International Conference on Educational Data Mining (EDM)*, pages 21–27, 2014.

- [11] Steve Ritter, Michael Yudelson, Stephen E. Fancsali, and Susan R. Berman. How mastery learning works at scale. In Jeff Haywood, Vincent Aleven, Judy Kay, and Ido Roll, editors, *Proceedings of the Third ACM Conference on Learning @ Scale (L@S)*, pages 71–79, 2016.
- [12] Oliver Scheuer and Bruce M. McLaren. Educational data mining. In Norbert M. Seel, editor, *Encyclopedia of the Sciences of Learning*, pages 1075–1079. Springer US, Boston, MA, 2012.
- [13] Kikumi K. Tatsuoka. Rule space: An approach for dealing with misconceptions based on item response theory. *Journal of Educational Measurement*, 20(4):345–354, 1983.
- [14] Michael V. Yudelson, Kenneth R. Koedinger, and Geoffrey J. Gordon. Individualized Bayesian knowledge tracing models. In H. Chad Lane, Kalina Yacef, Jack Mostow, and Philip Pavlik, editors, *Proceedings of the 16th International Conference on Artificial Intelligence in Education (AIED)*, pages 171–180, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

Neural Recall Network: A Neural Network Solution to Low Recall Problem in Regex-based Qualitative Coding

Zhiqiang Cai
University of
Wisconsin-Madison
zhiqiang.cai@wisc.edu

Cody Marquart
University of
Wisconsin-Madison
cody.marquart@wisc.edu

David W. Shaffer
University of
Wisconsin-Madison
dws@education.wisc.edu

ABSTRACT

Regular expression (regex) coding has advantages for text analysis. Humans are often able to quickly construct intelligible coding rules with high precision. That is, researchers can identify words and word patterns that correctly classify examples of a particular concept. And, it is often easy to identify false positives and improve the regex classifier so that the positive items are accurately captured. However, ensuring that a regex list is complete is a bigger challenge, because the concepts to be identified in data are often sparsely distributed, which makes it difficult to identify examples of *false negatives*. For this reason, regex-based classifiers suffer by having low recall. That is, it often misses items that should be classified as positive. In this paper, we provide a neural network solution to this problem by identifying a *negative reversion set*, in which false negative items occur much more frequently than in the data set as a whole. Thus, the regex classifier can be more quickly improved by adding missing regexes based on the false negatives found from the negative reversion set. This study used an existing data set collected from a simulation-based learning environment for which researchers had previously defined six codes and developed classifiers with validated regex lists. We randomly constructed incomplete (partial) regex lists and used neural network models to identify negative reversion sets in which the frequency of false negatives increased from a range of 3%-8% in the full data set to a range of 12%-52% in the negative reversion set. Based on this finding, we propose an interactive coding mechanism in which human-developed regex classifiers provide input for training machine learning algorithms and machine learning algorithms “smartly” select highly suspected false negative items for human to more quickly develop regex classifiers.

Keywords

Qualitative coding, Regex, text classification, neural network, recall, false negative density, negative reversion set

1. INTRODUCTION

Z. Cai, C. Marquart, and D. Shaffer. Neural recall network: A neural network solution to low recall problem in regex-based qualitative coding. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 228–238, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853047>

Data mining, as Baker (2010) described, is the “field of discovering novel and potentially useful information from large amounts of data”[2]. A critical challenge in *educational data mining* (EDM) is extracting information from text data. In education settings, text data might come from team chats in collaborative learning environments[4], conversations between computer agents and human learners[12], team discussions, notes, essays and reports in virtual internship[8, 10], etc. Traditionally, qualitative coding is the first and most important step in analyzing text data[11, 6]. But as the size of data increases, manual coding becomes expensive and in some cases impossible. Machine learning (ML) based text classification methods thus play an important role in EDM.

1.1 Challenges in Automated Text Classification

Unsupervised ML, such as *topic modeling*, can automatically extract collections of words that might serve as topics in text data[3, 5]. Supervised ML, such as neural network text classification algorithms[13, 1] can be trained on a subset of manually coded data to predict coding for the remainder of the data. While such ML algorithms can quickly code large data sets, they have disadvantages that may lead to biased classification. For example, unsupervised ML algorithms often generate topics that occur with high frequency. Codes with relatively low frequency are often missed or vaguely represented[7]. Supervised ML algorithms, on the other hand, rely on human coded data, thus bias in training data will be inherited by machine learning models. Moreover, in some circumstances, the amount of human-coded data required to produce a reliable classifier using supervised ML methods can be prohibitive.

A third approach to automated coding is to use human-developed *regular expressions* (regex). Humans are often able to quickly construct intelligible coding rules with high precision. That is, researchers can identify words and word patterns that correctly classify examples of a particular concept. It is often easy to identify *false positives* and improve the regex classifier so that the positive items are accurately captured. However, ensuring that a regex list is complete is a bigger challenge. Because the concepts to be identified in data are often sparsely distributed, examples of *false negatives* are difficult to identify. This causes the low-recall problem in regex based classifiers. That is, the regex based classifiers often miss items that should be classified as positive.

In this paper, we explore the possibility of constructing an interactive coding mechanism in which human-developed regex classifiers provide input for training machine learning algorithms and machine learning algorithms provide cues for human to construct better unbiased regex classifiers.

2. OVERVIEW OF CODING PROCESSES

2.1 Manual Coding

Manual coding of text data is a complex process. Done well, it requires researchers to intensively read their data to discover and construct appropriate theories and develop codes that reflect those theoretical machinery [11, 16, 15]. Hand coding typically begins when text data is segmented into meaningful units, or *items* of data. Codes are then induced by sorting and interpreting observations item by item. A grounded coding process iterates through three coding phases: *open coding*, *axial coding* and *selective coding*. In open coding, a researcher goes through the data to identify preliminary concepts. The concepts are then compared and categorized in axial coding. Once a conceptual framework is established, the researcher focuses on a fixed set of codes in the selective coding phase and identifies which items each code occurs.

Key to the process of manual coding is identifying a precise *definition* for each code (usually accompanied by examples from the data). However, coding depends on how individual researchers understand each code: there are no rules specified that convert text into code in a deterministic way.

To address this concern, manual codes are validated, either through a process of social moderation (sometimes call *dual coding*) in which two or more coders come to agreement about each code for each observation in the data, or using some form of *interrater reliability* (IRR) that quantifies the rate of agreement between two coders.

Because manual coding requires that at least one human rater codes each item of the data and includes triangulation with at least one more human rater, the results from manual coding are considered the most accurate (compared with machine coding methods) and often serves as the gold standard or *ground truth* to which other coding methods are compared.

Despite this advantage in terms of coding validity, manual coding can only be applied to data sets with relatively small size. Thus researchers turn to a variety of machine-based or machine-augmented coding techniques.

2.2 Keyword/Regex-based Coding

In the selective coding phase, when the size of a data set is too large for manual coding, researchers may turn to rule-based automatic coding [9]. Rules can be expressed in terms of text patterns described by regexes [14]. The simplest form of regex is a word, such as “teach”. If this word is used to represent a code “Education”, then the code “Education” occurs if and only if the text contains the string “teach”. Thus, if a text contains the words “teacher”, “teachers”, “teaches” or “teaching”, the code occurs, because “teach” is a sub-string of all these words. Regexes use a set of operators to specify complex patterns. The most frequently used operators

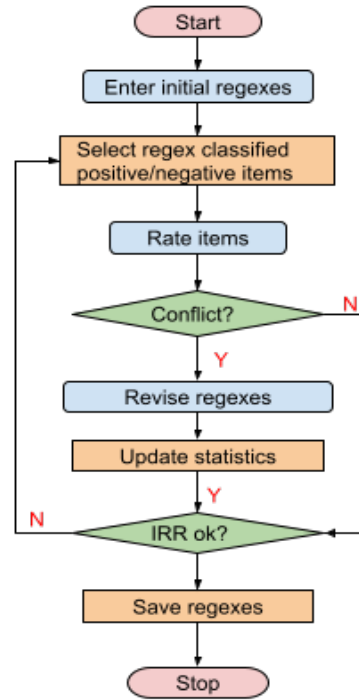


Figure 1: nCoder flowchart.

include the pipe symbol “|”, period “.”, brackets “[]”, and star “*”. The pipe symbol “|” is used to compose alternative strings. For example, the regex “teach|student|school” matches a text containing any of the strings “teach”, “student” or “school”. A period “.” indicates an arbitrary text character and brackets “[]” indicate a range of characters (e.g., [aeiouy] indicates any vowel in English), and the star “*” indicates an arbitrary repetition of the previous character. So the regex “my favorite vowel.*[aeiouy]” would code for the text “my favorite vowel” followed anywhere in the text string by a vowel. In this way, very complicated patterns can be represented by a single regex(see, e.g., <https://www.regular-expressions.info/tutorial.html>).

One powerful publicly available tool for regex-based coding is nCoder (<https://app.n-coder.org/>). nCoder uses an *active machine learning* approach to generating a regex-based classifier. Figure 1 shows the typical work flow of nCoder. A researcher starts from an initial regex list. nCoder uses that regex list to select a set of items to present to the researcher for rating. The set may proportionally include items that the regex list classified as positive or negative. nCoder then allows the researcher to rate the items and compares the researcher’s rating with the regex classification. When conflict occurs, the researcher may revise the regex list, change his or her rating, or leave the conflict unresolved. Inter-rater reliability (IRR) between the researcher and the regex classifier is then computed based on hypothesis testing. nCoder may present another set of items and repeat the rating-testing process. The process continues until the researcher and regex classifier reach a satisfactory level of agreement. The developed regexes are saved before the process terminates.[16].

The convergence of the above procedure depends on what data items are presented to the researcher. If a presented item is one that the human and regex classifier have agreement, no information is provided to the human to improve the regex list. Rather, it is the items where the two ratings conflict that help improve the regex list. This conflict may occur in two ways: one is *false positive*, in which the regex coded positive but the human coded negative; the other is *false negative*, in which the regex coded as negative but the human coded as positive.

Conceptually, we can partition a data set D into two sets: a positive set $\tilde{P} \subset D$ of the items that would be coded as positive by current regex, and a negative set $\tilde{N} = D - \tilde{P}$ that would be coded as negative by the regex list. False positive items are a subset of the positive set \tilde{P} (because a false positive requires that the regex codes an item as positive); and false negative items are only in the negative set \tilde{N} (by similar reasoning).

Notice, however, when the frequency of a code is low (that is, there are few items that should be coded positive), the probability of sampling a false positive item from the positive set \tilde{P} may be much higher than sampling a false negative item from the negative set \tilde{N} . Thus, it is more difficult for the active machine learning system to present users with false negatives, potentially allowing the regex development procedure to converge on a regex list with high precision but potentially lower recall (see the definition of precision and recall in section 3.4).

2.3 Machine learning-based Coding

Machine learning algorithms can be applied to both code discovery and selective coding. In the domain of code discovery, unsupervised machine learning algorithms, such as topic modeling, can automatically extract from a data set lists of related words that might potentially serve as codes. However, research shows that in most cases these word lists do not produce codes of interest to qualitative researchers without further refinement. But topic modeling *can* help identify missing codes by providing potential keywords for coding[5].

One popular approach to using ML for selective coding is LSTM (Long Short Term Memory) neural network models. These models take as input a set of coded data items, and return an algorithm that can predict coding for the rest of the data. Figure 2 shows the model used in this study. A text data item enters from the input layer as an ordered word list. The embedding layer represents each unique word by a vector. The ordered vectors are input to a bidirectional LSTM layer that creates two vector representations of the input data line. The two vectors are merged into a single vector representation and 50% of the nodes are removed in the dropout layer to control over-fitting. The sigmoid layer converts the vector into a class probability output, which indicates for each item how likely the model thinks it should be coded positive. The probability values are then converted to a binary coding value using a cutoff threshold, say, 0.5. That is, when the probability is greater than 0.5, the item is coded as 1, otherwise 0. (Readers interested in more details about LSTM models can find more information in neural network publications such as [13].)

Table 1: Manual, Regex and ML Coding

	Manual	Regex	ML
Evidence	Explicit	Explicit	Implicit
Precision	High	High	Medium
Recall	High	Low	Medium
Coding size	Limited	Unlimited	Unlimited
Cost	High	Low	Medium

This kind of neural network model can represent very complicated patterns of words that serve as indicators for a code. However, it usually requires a large number of coded data items to train a neural network model to reach an acceptable accuracy; thus preparing the training data for the neural network model could be expensive. Moreover, like many other machine learning models, neural network models are a *black box*: the output is hard to interpret. When a neural network model flags the occurrence of a code in a data item, it can be hard (and often impossible) for a researcher to determine what specific evidence the algorithm used to make its decision. This makes it difficult for qualitative researchers to establish a theory with clear interpretation. In addition, it makes it difficult for researchers to understand potential biases in the codes and to warrant to end users that the result of the coding is fair.

2.4 Combining manual, regex and ML coding

Table 1 summarizes the advantages and disadvantages of the three coding methods we have discussed so far. In what follows, we construct and test an approach to coding that integrates these three methods together into a single process, in which we iterate between manual coding and regex coding to train a regex list. Then we use the data coded by the regex list to train the neural network model. The disagreements between regex coding and neural network coding are presented back to the researcher to further refine the regex coding. We will show that this combination helps solve the low recall problem in regex coding.

2.5 Approach

For a given data set D and a code c , let a human rater’s classification be $D = P + N$, where P is the set of positive items (i.e., items in which code c occurs) and N is the set of negative items. The plus sign “+” denotes the union of two sets. In practice, the number of items in P is usually much smaller than in N (i.e., $|P| \ll |N|$), because researchers are often interested in codes that do not occur with high frequency.

Of course, for a given item $x \in D$, the coding algorithm cannot determine whether $x \in P$ or $x \in N$ without asking the human rater. Therefore, it is impossible for an algorithm to sample a new item from P (or N) unless they have already been classified by the human rater. In this sense, the sets P and N are *unsamplable*, while the set D is *samplable*. That is, a machine algorithm can choose an item $x \in D$ but it cannot choose with certainty whether sample $x \in P$ or $x \in N$.

Consider a regex classifier that classifies the data set $D = \tilde{P} + \tilde{N}$. Since the machine knows the classification of all

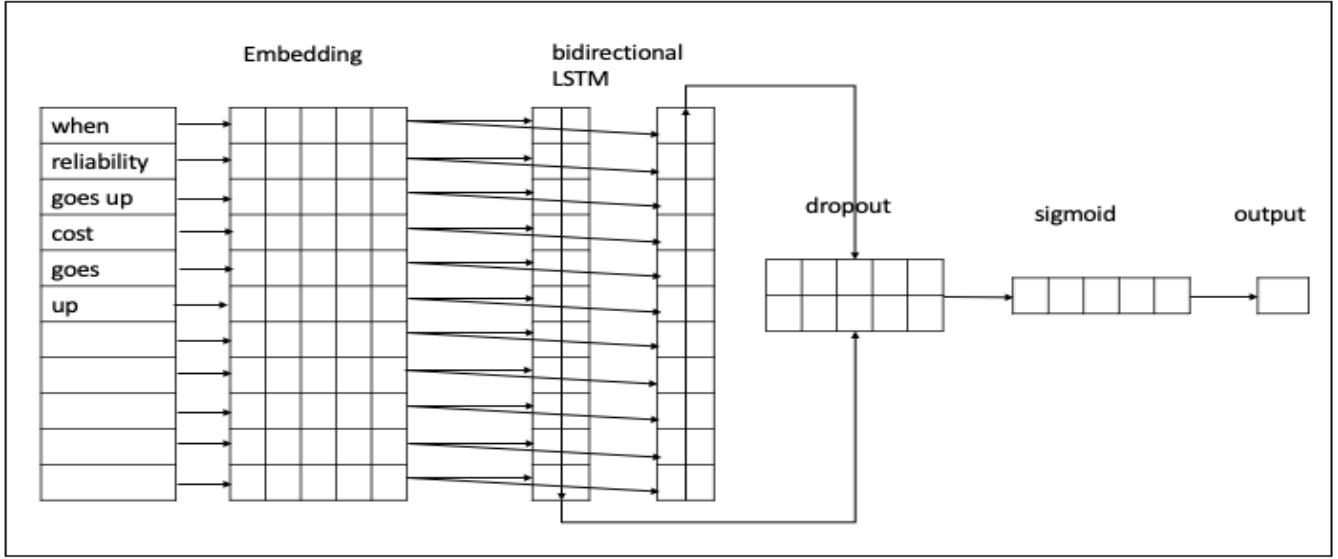


Figure 2: LSTM neural network model for text classification

items in D , the sets \tilde{P} and \tilde{N} are samplable. The regex classifier is perfect if $P = \tilde{P}$. In reality, however, this is rarely the case.

The data set D thus can be decomposed as the union of four sets (from now on we use the notion XY to denote the intersection of the sets X and Y):

$$D = P\tilde{P} + P\tilde{N} + N\tilde{P} + N\tilde{N}$$

where

- $P\tilde{P}$: true positive set of the regex classifier relative to human;
- $P\tilde{N}$: false negative set of the regex classifier relative to human;
- $N\tilde{P}$: false positive set of the regex classifier relative to human; and
- $N\tilde{N}$: true negative set of the regex classifier relative to human.

Notice that, since P and N are unsamplable, none of the above sets ($P\tilde{P}$, $P\tilde{N}$, $N\tilde{P}$, or $N\tilde{N}$) is samplable.

As described above, the performance of the regex classifier development process depends on its ability to present users with false positives $N\tilde{P}$ and false negatives $P\tilde{N}$. When $\tilde{P} \ll \tilde{N}$, the false negative items are less *recoverable*, in the sense that the probability of sampling a false negative item is much smaller than false positive.

Let us use an example to show this. Suppose the data size is $|D| = 10,000$ and a regex classifier identified 1000 positive items $|\tilde{P}| = 1000$ with 500 false positive and 500 false negative: $|N\tilde{P}| = 500$ and $|P\tilde{N}| = 500$. Since the false positive

items are in the samplable set \tilde{P} , we can sample from \tilde{P} to find false positive items. The probability of sampling a false positive item from \tilde{P} is $500/1000 = 0.5$. On the other hand, false negative items are in the samplable set \tilde{N} , which contains $10000 - 1000 = 9000$ items. The probability of sampling a false negative item from \tilde{N} is $500/9000 \approx 0.056$.

The above example shows that, when developing a regex classifier, it is hard for the machine to find false negatives — that is, to find examples of items that the human would classify as positive but the regex classifier does not. This means that positive cases outside of the scope of the current regular expression list are difficult for the regex classifier to identify, which causes the low recall problem.

Here we ask whether it is possible to use a neural network model trained from the regex coded data to identify false negative items. For example, if a regex list for the code “education” contains “teach”, “student” and “school”, a neural network model may be able to tell that the items containing “class”, “book” should be included in the positives. Thus, if we sample items that the regex classified as negative but neural network model classified as positive, we may be able to increase the probability of finding examples that the human rater would code as positive but the regex classifier would not.

Consider a neural network model that classifies $D = \tilde{\tilde{P}} + \tilde{\tilde{N}}$. The regex false negative set $P\tilde{N}$ is decomposed by the neural network classifier as

$$P\tilde{N} = P\tilde{\tilde{P}} + P\tilde{\tilde{N}}.$$

We are mostly interested in the items that regex classifies as negative but neural network model classifies as positive. We call the set of such items *Negative Reversion Set*, which

can be written as

$$\tilde{N}\tilde{P} = P\tilde{N}\tilde{P} + N\tilde{N}\tilde{P}.$$

Notice that the negative reversion set $\tilde{N}\tilde{P}$ is the intersection of the regex negative set (items coded negative by the regex) \tilde{N} and the neural network positive set (items coded positive by the neural network) \tilde{P} . Since both \tilde{N} and \tilde{P} are samplable, the negative reversion set $\tilde{N}\tilde{P}$ is samplable.

2.6 Research question

In what follows, we ask two research questions:

2.6.1 RQ1

Is the probability of finding a regex false negative item higher from the negative reversion set $\tilde{N}\tilde{P}$ than from the regex negative set \tilde{N} ? In other words, are the false negatives denser in the negative reversion set $\tilde{N}\tilde{P}$ than in the regex negative set \tilde{N} ?

We define the false negative density d_A of a set A as the proportion of the number of false negative items in the set A . That is,

$$d_A = \frac{|P\tilde{N}A|}{|A|}.$$

Thus, the false negative density of a negative reversion set $\tilde{N}\tilde{P}$ is

$$d_{\tilde{N}\tilde{P}} = \frac{|P\tilde{N}\tilde{P}|}{|\tilde{N}\tilde{P}|},$$

and the false negative density of a regex negative set \tilde{N} is

$$d_{\tilde{N}} = \frac{|P\tilde{N}|}{|\tilde{N}|}.$$

Using the notation of false negative density, the mathematical form of this research question is whether or not the following equation holds:

$$d_{\tilde{N}\tilde{P}} \gg d_{\tilde{N}} \quad (1)$$

2.6.2 RQ2

If a set A is found dense with false negatives, does it also has an acceptable false negative recovery rate?

Sampling false negatives in a denser set certainly helps. However, it is also important that the denser set should be large enough to include a certain proportion of false negatives. We define the *false negative recovery rate* r_A of a set A as the proportion of false negatives included in the set A , that is

$$r_A = \frac{|P\tilde{N}A|}{|P\tilde{N}|}. \quad (2)$$

So, we ask whether or not the false negative recovery rate of the negative reversion set $r_{\tilde{N}\tilde{P}}$ is large enough.

Table 2: Code definition.

Code	Description
CONSTRAINTS	Referring to inputs: material, processing method, surfactant, and CNT.
PERFORMANCE	Referring to attributes: flus, blood cell reactivity, marketability, cost, or reliability.
COLLABORATION	Facilitating a joint meeting or the production of team design products.
DESIGN	Referring to design and development prioritization, tradeoffs, and design decisions.
DATA	Referring to or justifying decisions based on numerical values, results tables, graphs, research papers, or relative quantities.
REQUESTS	Referring to or justifying decisions based on internal consultant’s requests or patient’s health or comfort.

2.6.3 General method

To answer these questions, we investigate a text data set with full regex lists developed and validated for multiple codes. The full regex lists coded data is used as “true human ratings”. We then randomly sample a partial regex list from the full regex list of each code. Partial regex lists are treated as the regex classifiers under development. Neural network models are built from the data coded by these partial regex lists. So, given this data set, we are able to obtain “human ratings” (actually full regex list coding), regex classifier (actually partial regex classifier) and neural network classifier (trained on partial regex classifier). The negative reversion sets are then determined from these three classifications. We are then able to compute the false negative density and the false negative recovery rate of the obtained negative reversion sets to get the answers to our research questions.

3. METHODS

3.1 Data

The data used in this study consists of 50,818 participant utterances collected from novice engineering design teams participating in an engineering virtual internship *Nephrotext*, in which students worked as interns at a fictitious company that designs and manufactures ultrafiltration membranes for hemodialysis machinery used to treat end-stage renal failure[8]. Table 2 shows the definition of six codes defined by previous researchers, including *Tech Constraints*, *Performance*, *Collaboration*, *Design*, *Data* and *Requests*.

Using nCoder tool (<https://app.n-coder.org/>), researchers developed and validated the regex lists. Table 3 shows the regex list for each code, together with “base rate” (BR) and kappa values. The base rate of a code was the proportion of items in the data set the code occurred. The kappa values

Table 3: Regex lists for each code, the base rates (BR), and the kappas between human rater 1 (H1) and human rater 2 (H2), human rater 1 and computer (C) and human rater 2 and computer.

Code	Regex	BR(%)	H1-H2	H1-C	H2-C
CONSTRAINTS	\bPESPVP, \bdry-jet, \bnegative charge, \bsurfactant, \bchemical, \bvapor deposition polymerization, \bjet \bPMMA, \bPRNLT, \bmanufacturing process, \bmaterials, \bphase inversion, \bvapor, \bsteric, \bPolyamide, \bnano, \bbiological, \bprocesses, \bpolysulfone, \bhydro, \bcarbon nanotube, \bCNT	16.086	0.96	1.00	0.96
PERFORMANCE	\bafforda, \bBCR, \bflux, \bexpensive, \bmarketa, \bcharacteristic, \bcatagories, \bsafe, \bprice, relia, \bblood cell, bility, \bcost	14.508	0.88	0.93	0.84
COLLABORATION	\bmeeting, \bwe all, \bdiscussion, \bwhat should, \beverybody, \bwe could, \bdo we, \bteammates, \bshar, (.*?\bpeople.*?\bteam.*?), (.*?\bteam.*?\bpeople.*?), \bwe should, \bsuggesting, \bshould we	8.861	0.76	0.87	0.76
DESIGN	\bfinal decision, \bdecision, \bwent with, \bbased each design, \bbalance, \bsacrifice, \bsay the first, \bpick, \bI think we should, \bwe had to find, \bbalance, \bmaybe use, \bwe could, \bare we doing, \bcompromise, \bwe changed, \bstick with, \bchoosing, \bdecide, \bliked best, \bbalancing, \bshould we raise, \bimprove the design, \bIm trying that one, \bhow did you design, \bchoose, \bwe want to design, (.*?\bincrease.*?\bdesign.*?), (.*?\bdesign.*?\bincrease.*?), \btargeted, (.*?\bcould be.*?\bdesign.*?), (.*?\bdesign.*?\bcould be.*?), \bI would say, \bwe raised, \bdecrease, \bcomparing, \bsuperior, (.*?\bwe could.*?\bdesign.*?), (.*?\bdesign.*?\bwe could.*?), (.*?\bbest.*?\boption.*?), (.*?\boption.*?\bbest.*?), \bchose \beither way, \blet's go with, \bwe do it like, \bchoice, (.*?\bI think.*?\bbest.*?), (.*?\bbest.*?\bI think.*?), \bwe can do, (.*?\bI think.*?\bsubmit.*?), (.*?\bsubmit.*?\bI think.*?), \bdo we want, \btradeoff, \bbest way, \bmanipulated, \bminimize, \btrade off \bchose	10.291	0.89	0.86	0.84
DATA	(.*?\blowest.*?\bcheapest.*?), (.*?\bcheapest.*?\blowest.*?), \bperformed well, \bmaximizes, \bhad great reliability, \bresult, \brates, \bscore, \b(?<!player)(?<!player)(?<![])(?<!:[1-9][0-9](?!%)(?! %)(?!min)(?!min)(?!:)(?!pm)(?!am), \bhad the lowest reliability \bscore, (.*?\bseems to be.*?\bcostly.*?), (.*?\bcostly.*?\bseems to be.*?), \bworst, \bpoor, \bchart, \bequal value, \bresults, \bwas found to be, \breeding, \btoo high, (.*?\bperformed.*?\buniformly.*?), (.*?\buniformly.*?\bperformed.*?), \bperform well, \baverage, \btests, \bcost more, \bwas good in, \bgraph, \bgraph, \brates, \bdata, \bperforms.*?\breliability, (.*?\boverall.*?\bperformed.*?), (.*?\bperformed.*?\boverall.*?)	10.405	0.94	0.9	0.89
REQUESTS	\buser, \bDuChamp, \bPadma, \bsafety, \bhospital, \bstandard, \bcomfort, \brecommendations, \bRudy, (?:(?!*\bexternal).)*\bconsultant(?!*\bexternal), \bMichelle, \bminimum, \bWayne, \brequest, \bsatisfy, \bpatient, \bProctor, \binternal consultant, \bHernandez, \brequirement \bAnderson, \bunacceptables, \bclient, \bAlan, \bRao	7.059	0.88	0.94	0.94

were computed on the ratings from two human raters (H-1, H-2) and the regex classifier (C). Because of the high kappa values, we considered these regex lists as “complete” and represent the “true classification” by human raters. In the discussions below, we use “full regex classification” to replace “human classification”.

We decomposed the items in the regex lists if they were connected by a pipe symbol “|”. For example, if an item was “r1|r2”, then it was decomposed as two items “r1” and “r2”. Of course, all items in the list for a given code could be composed back as a single regex by re-connecting them with a pipe symbol “|”. We decomposed the regex in this way so that we could more meaningfully sample partial regex lists.

3.2 Data splitting

The data set D was randomly split into a training set S and a test set T , each of which consisted of 25,409 items. The training set S was used for sampling items to train neural network models, while test set T was used for computing final results.

3.3 Coding by three classifiers

3.3.1 Full regex coding

Full regex lists for the six codes were used to code the whole data set D , which formed the “true classification” of the whole data $D = P_D + N_D$. Consequently, the training set S and the test set T are also “truly” classified as $S = P_S + N_S$ and $T = P_T + N_T$. Under this classification, the positive rates (base rates) for the six codes range from 7% to 16% (see Table 3).

3.3.2 Partial regex coding

For a given code, a partial regex list was constructed using the following procedure:

1. Start with an empty regex list $L = \{\emptyset\}$ and full regex list R_c for a code c ;
2. Randomly select a regex $r \in R_c$ and add it to the partial regex list L ;
3. Code the whole data set D by the partial list L to produce $\tilde{P}_{D/L}$;
4. Compute the number of positive items $|\tilde{P}_{D/L}|$ coded by L ;
5. If $|\tilde{P}_{D/L}| > \frac{|P_D|}{2}$, end the procedure; otherwise go back to step 2.

Through this procedure, the number of positive items classified by a partial regex list L is equal to or greater than half of the true positive items. Each partial list L decomposed the training set and test set into the following samplable sets (for parsimony, we define $\tilde{P}_A \equiv \tilde{P}_{A/L}$):

- \tilde{P}_S : regex positives in training set;
- \tilde{P}_T : regex positives in test set;
- \tilde{N}_S : regex negatives in training set; and
- \tilde{N}_T : regex negatives in test set.

3.3.3 Neural network coding

In this paper, the neural network models were trained based on the classification of partial regex classifiers. For a given partial regex classifier and a given sample size n , a random sample $s_n \in S$ with size n was taken from the training set. Each item in the sample contained two fields: the text field X and the classification value Y (1 when the regex list was matched and 0 otherwise). The text field X is used as the input and the partial regex classification value Y is used as the output to train a predictive LSTM neural network model. The testing set was then coded by the predict function of the LSTM model, with the cutoff probability 0.5. Thus, the neural network model classified the test set into the samplable sets:

- \tilde{P}_T : neural network positive in test set; and
- \tilde{N}_T : neural network negative in test set.

3.3.4 Sample size and repetition of partial regex lists

In training the neural network models, we used five different sample sizes: $n = 100, 200, 400, 800, 1600$. In order to reduce the random effect in partial regex construction, 12 random partial regex lists were drawn for each code using the partial regex construction procedure. So, for each of the six codes, the test set had the following classifications:

- 1 “true classification” $T = P + N$;
- 12 partial regex list classification $T = \tilde{P}_i + \tilde{N}_i$, ($i = 1, 2, \dots, 12$);
- 60 neural network classification (12 partial regex list, 5 sample sizes each) $T = \tilde{P}_{in} + \tilde{N}_{in}$, ($i = 1, 2, \dots, 12; n = 100, 200, 400, 800, 1600$).

In the notions above, \tilde{P}_i and \tilde{N}_i were the positive and negative set, respectively, classified by the i^{th} partial regex for the given code; and \tilde{P}_{in} and \tilde{N}_{in} were the positive and negative set, respectively, classified by the neural network model trained on the i^{th} partial regex list with sample size n .

3.4 Performance metrics

Before moving on, we define three measures for the performance of classifiers: *precision*, *recall* and *Cohen’s κ* . For a given classifier, denote the *proportion* of true positives, false positives, false negatives and true negatives by tp, fp, fn and tn , respectively.

- The **precision** of a classifier is the ratio of true positive to the sum of true positive and false positive, namely

$$precision = \frac{tp}{tp + fp}.$$

- The **recall** of a classifier is the ratio of true positive to the sum of true positive and false negative, namely

$$recall = \frac{tp}{tp + fn}.$$

- The **Cohen’s kappa** κ of a classifier is the ratio of the difference between observed probability and chance probability to one minus chance probability, namely,

$$\kappa = \frac{p_o - p_c}{1 - p_c}.$$

where

$$p_o = tp * fp,$$

and

$$p_c = fp * fn + (1 - fp) * (1 - fn).$$

4. RESULTS

4.1 Single model comparison

Table 4 shows an example classification on code “Tech Constraints” with its full regex list, one randomly constructed partial regex list and one neural network model trained from the partial regex list with sample size 400.

In this example, the classification of the test set by the full regex classifier, which was considered the “true classification”, yielded 3976 positive items (set P) and 21,431 negative items (set N). Notice that there were far more negative items than positive items. The partial regex classifier correctly identified 2,133 positive items (set $P\tilde{P}$). However, it falsely classified 1845 positive items as negative (set $P\tilde{N}$). Since the partial regex was a subset of the full regex list, any item matched by the partial regex was in turn matched by the full regex. In other words, any item not matched by the full regex was not matched by the partial regex. Thus, no negative item was falsely classified as positive (set $N\tilde{P}$); and the partial regex classifier agreed with the full regex classifier on all 21,431 negative items (set $N\tilde{N}$). As a result, the partial regex got a larger negative set $\tilde{N} = P\tilde{N} + N\tilde{N}$ with a total of 23,276 items. The false negative density in the negative set \tilde{N} was thus $d_{\tilde{N}} = 1845/23276 \approx 7.92\%$.

The third column of the table shows the classification results of the neural network model trained on the data classified by the given partial regex list with sample size 400. There were 1170 items that were correctly classified by both partial regex list and the trained neural network model (set $P\tilde{P}\tilde{\tilde{P}}$). 963 positive items were correctly classified by the partial regex list but falsely classified as negative by the neural network model (set $P\tilde{P}\tilde{N}$). Most interestingly, there were 288 positive items that were falsely classified as negative by the partial regex list but correctly reversed as positive by the neural network model (set $P\tilde{N}\tilde{\tilde{P}}$). 1557 positive items were falsely classified as negative by both the partial regex list and neural network model. For the 21,431 negative items, the neural network falsely classified 302 items as positive (set $N\tilde{N}\tilde{\tilde{P}}$) and correctly classified 21,129 items as negative (set $N\tilde{N}\tilde{\tilde{N}}$).

The negative reversion set $\tilde{\tilde{N}}$ was the union of the sets $P\tilde{N}\tilde{\tilde{P}}$ and $N\tilde{N}\tilde{\tilde{P}}$, which contained 288 correctly reversed items and 302 falsely reversed items. Therefore, the false negative density in the negative reversion set $\tilde{\tilde{N}}$ was $d_{\tilde{\tilde{N}}} = 288/(288 + 302) \approx 48.81\%$, which was much larger than the density in the regex negative set $d_{\tilde{N}} = 7.92\%$.

Table 4: An example of false negative recovery for Tech Constraints with training size 400.

Full Regex	Partial Regex	Neural Network
$P(3976)$	$P\tilde{P}$ (2133)	$P\tilde{P}\tilde{\tilde{P}}$ (1170)
	$P\tilde{N}$ (1845)	$P\tilde{P}\tilde{N}$ (963)
		$P\tilde{N}\tilde{\tilde{P}}$ (288)
		$P\tilde{N}\tilde{\tilde{N}}$ (1557)
$N(21,431)$	$N\tilde{P}$ (0)	$N\tilde{P}\tilde{\tilde{P}}$ (0)
	$N\tilde{N}$ (21,431)	$N\tilde{P}\tilde{N}$ (0)
		$N\tilde{N}\tilde{\tilde{P}}$ (302)
		$N\tilde{N}\tilde{\tilde{N}}$ (21,129)

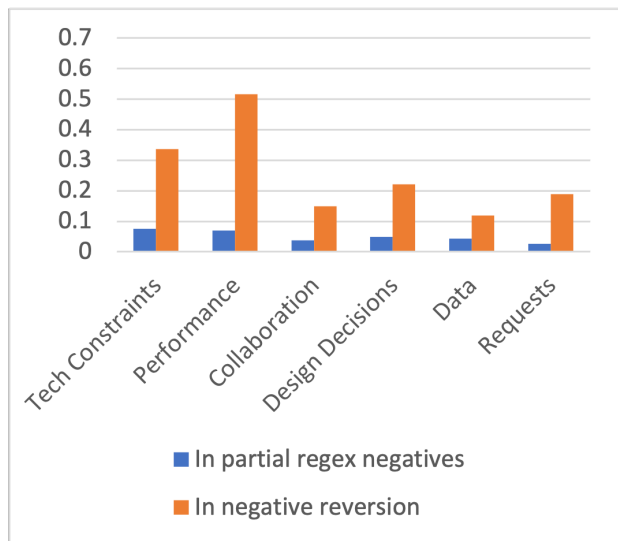


Figure 3: False negatives density in partial regex negatives versus in negative reversion set.

This example gives us a positive answer to our first research question. That is, the false negative density in the negative reversion set is much higher than in the regex negative set.

To answer the second question, we computed the false negative recovery rate in the negative reversion set. From Equation 2, we had $r_{\tilde{\tilde{N}}} = 288/1845 \approx 16\%$.

If our goal is to find all false negatives, this number is not very high, because 84% false negatives are still sparsely distributed in the regex negative set \tilde{N} . However, we argue that this number is large enough for constructing iterative methods, which we will talk more about in the discussion section.

4.2 Average false negative density

The above example showed false negative density and the false negative recovery rate in the negative conversion set for one code, one random partial regex list and one neural network model with sample size 400. To answer our research questions more reliably, we computed the average false negative density for each code with 12 randomly drawn partial

Table 5: False negative recovery rate, training size=400

Code	$P\tilde{N}\tilde{P}$	$P\tilde{N}\tilde{N}$	r (%)
Tech Constrains	195.83	1579.67	11.03
Performance	270.17	1338.00	16.80
Collaboration	23.75	900.58	2.57
Design	76.67	1908.17	6.53
Data	19.67	1027.67	1.88
Requests	38.25	615.25	5.85

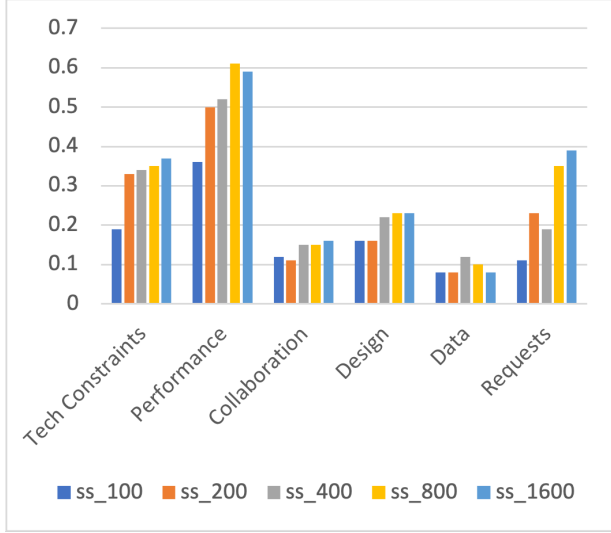


Figure 4: False negative density in negative reversion set for all sample sizes.

regex lists. The results show that the false negative density for all codes were several times higher in the negative reversion set $\tilde{N}\tilde{P}$ than in the regex negative set \tilde{N} . Figure 3 shows the results for sample size=400. While the density $d_{\tilde{N}}$ in the regex negative set ranged from 3% to 8%, the density $d_{\tilde{N}\tilde{P}}$ in the negative reversion set ranged from 12% to 52%. For example, the density for the code “Performance” increased from $d_{\tilde{N}} = 7\%$ to $d_{\tilde{N}\tilde{P}} = 52\%$.

4.3 Average false negative recovery rate

For each code, we computed the average false negative recovery rate over 12 randomly drawn partial regex lists. Table 5 shows, with neural network training size=400, the number of items in sets $P\tilde{N}\tilde{P}$ and $P\tilde{N}\tilde{N}$ and false recovery rate as percentages. The average false negative recovery rates ranged from 1.88% to 16.88% and the average number of false negative items included in the negative reversion set ranged from 19.67 to 270.17.

4.4 Sample size effect

In the above, we only showed the results for the neural network models trained from sample size $n = 400$. Figure 4 and Figure 5 show the average false negative density and false negative recovery rate in the negative reversion set for each code as a function of neural network training size. For the codes “Tech Constraints”, “Collaboration” and “Design”,

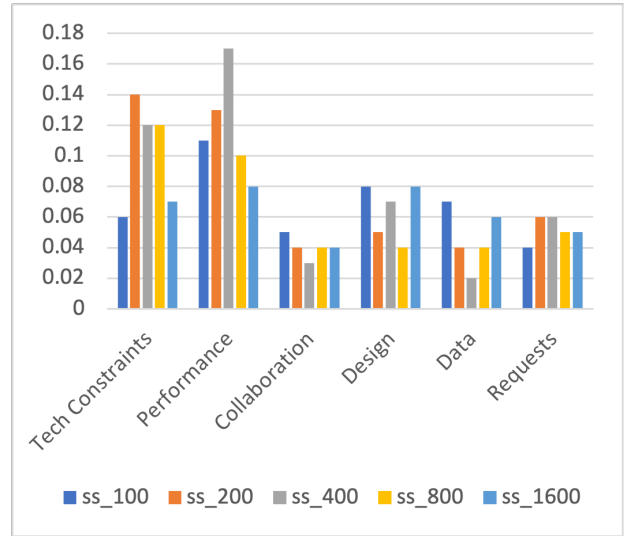


Figure 5: False negative recovery rate for all sample sizes.

the density change was very small after size 400. For code “Performance” and “Requests”, the density for larger training size was larger. However, they suffered from reduced false negative recovery rate.

We also computed Cohen’s kappa between different models. Figure 6 shows the mean Cohen’s kappas for each code as functions of sampling size for neural network model. The kappa means were computed over the 12 partial regex lists for each code. The blue lines are the kappas between full regex and partial regex lists. They are horizontal lines because they have nothing to do with the sample size. The green lines are kappas between partial regex lists and neural network models. As sample size increases, the green lines increases. That indicates that neural network models could be close to the partial regex list model as the sample size increases. For codes “Tech Constraints” and “Performance”, the green lines even go beyond the blue lines when the sample size is large. This is normal because the neural network models were trained from partial regex list. The red lines are kappas between the full regex and neural network models. Although the red lines also increase, they never go beyond the blue lines. That indicates that, although neural network models may help identify items missed in partial regex list, they don’t perform better than the regex classifier from which they are trained.

5. CONCLUSIONS

In this study, we used neural network models trained from partial regex classifier to help identify the false negatives from partial regex classifiers. The so called *negative reversion* set $\tilde{N}\tilde{P}$ was of our most interest in this study. This set consisted of conflict items that the partial regex classifier coded as negative but the neural network classifier coded as positive. Since the neural network classifiers was trained from the data coded by the partial regex classifiers, the negative reversion sets were actually error sets - the *false positive* sets relative to the partial regex coded data. That is, the neural network models didn’t correctly predict how the par-

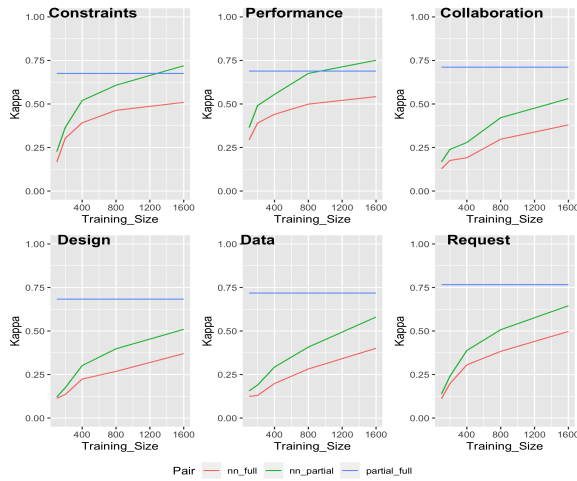


Figure 6: Kappa as functions of training size between three pairs of coding: partial regex and full regex coding, neural network and full regex coding, and neural network and partial regex coding.

tial regex classifiers code such items. However, our study showed that this “error set” had a much higher density of regex false negatives. This indicates that the neural network models had some ability in detecting the error contained in the training data. It is unclear how this happened. Our current theory is that, since the neural network model was built on semantic representations (word embedding layer captures the word meaning and the LSTM layer captures the word order information), it could be able to detect certain violations of semantic consistency. For example, if “table” and “chair” are included in a partial regex list, the neural network model may suggest that “furniture” shouldn’t be excluded.

The results show the differences of the density increases among codes. The increased density in the false negative reversion set doesn’t correlate to the original density in the regex false negative set. The source of the difference is unknown. However, the fact that the density in the false negative reversion set is from 3 to 12 times higher suggests that this technique could improve classifier performance.

The results showed that false negative recovery rate ranged from 2% to 17%, depending on the specific code and neural network training size. This indicates that, while it is easier to find false negative items from the negative reversion set, many false negative items remain in other sets. We suggest that, when searching for false negative items, the negative reversion set and the whole regex negative set should both be considered.

The training size issue in this study is more complex. In general, more training data results in better neural network models. However, in our case, we are expecting a larger “error set” $\tilde{N}\tilde{P}$. When the training size is too large, the “error set” shrinks and thus reduces the false negative recovery rate. From our study, it appears that a training size of approximately 400 is adequate.

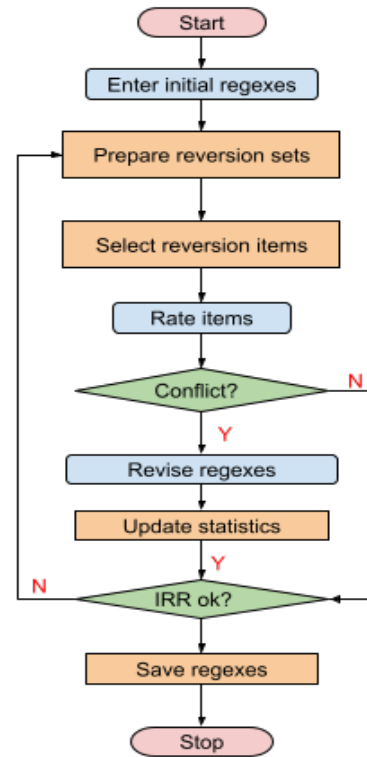


Figure 7: Neural network assisted regex development.

More generally, our study is limited by the use of one specific data set, six specific codes, and a specific ratio (1:2) between the partial regex and the full regex. In practice, the partial regex corresponds to the regex list under development and the full regex corresponds to the human understanding of the code. When the partial regex is close to the “full regex”, the density in the negative reversion set may become small.

While, of course, training the neural network models from manually coded data would produce a better predictor, it would require a human rater to code a large number of items. But, as we argued above, even if enough manual coding could be provided, training a neural network model directly from the manually coded data will result in a classifier that is hard to interpret and defend. Instead, we suggest using an imperfect neural network model to augment regex-based classification, so that when a regex classifier codes a positive item, it will be clear to researchers and end users why each item has been coded.

The last but not least, we didn’t use real human coding in this study, which limited our investigation on false negative items only, because a partial regex classifier will never have false positive prediction in relation to a full regex classifier. However, in the case of real human coding, false positives are likely to occur. Similar to the negative reversion set, we may define a positive reversion set as the set of items for which the regex classifier coded as positive but the human rater coded as negative. It could be the case that the positive reversion set also contains denser false positives. However, this is less important when the positive rate is small.

To conclude this paper, we propose the following iterative procedure for better development of regex classifiers (see Figure 7):

1. The user enters initial regex list for development;
2. A machine learning model is trained based on the initial regex list classification and prepares reversion sets;
3. The computer selects highly likely conflict items suggested by machine learning model;
4. The user rates the item ;
5. If a conflict occurs, the user resolves the conflict and updates the regex list;
6. The computer computes agreement statistics (IRR) based on the ratings from the user, regex list and neural network model;
7. If the statistics show that the IRR is not high enough, go back to the reversion set preparation step and reiterate the process;
8. If the statistics show high agreement, end the process.

6. ACKNOWLEDGMENTS

This work was funded in part by the National Science Foundation (DRL-1713110, DRL-2100320, LDI-1934745), the Wisconsin Alumni Research Foundation, and the Office of the Vice Chancellor for Research and Graduate Education at the University of Wisconsin-Madison. The opinions, findings, and conclusions do not reflect the views of the funding agencies, cooperating institutions, or other individuals.

7. REFERENCES

- [1] X. Bai. Text classification based on lstm and attention. In *Thirteenth International Conference on Digital Information Management (ICDIM)*, pages 29–32, 2018.
- [2] R. Baker. Data mining for education. *International encyclopedia of education*, 7(3):112–118, 2010.
- [3] D. M. Blei and A. Y. Ng. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3(4-5):993–1022, 2003.
- [4] Z. Cai, B. Eagan, N. M. Dowell, J. W. Pennebaker, D. W. Shaffer, and G. A. C. Epistemic network analysis and topic modeling for chat data from collaborative learning environment. In *Proceedings of the 10th International Conference on Educational Data Mining*, pages 104–111, 2017.
- [5] Z. Cai, A. Siebert-Evenstone, B. Eagan, and D. W. Shaffer. Using topic modeling for code discovery in large scale text data. In *Advances in Quantitative Ethnography: ICQE Conference Proceedings*, pages 18–31, February 2021.
- [6] K. Charmaz. *Constructing grounded theory*. Sage, London, 2006.
- [7] N.-C. Chen, M. Drouhard, R. Kocielnik, J. Suh, and C. R. Aragon. Using machine learning to support qualitative coding in social science: Shifting the focus to ambiguity. *ACM Trans. Interact. Intell. Syst.*, 8(2):9:1–9:20, June 2018.
- [8] N. Chesler, A. Ruis, W. Collier, Z. Swiecki, G. Arastoopour, and D. Shaffer. A novel paradigm for engineering education: virtual internships with individualized mentoring and assessment of engineering thinking. *Journal of Biomechanical Engineering*, 137(2):1–8, 2015.
- [9] K. Crowston, X. Liu, E. Allen, and R. Heckman. Machine learning and rule-based automated coding of qualitative data. In *Proceedings of ASIST 2010*, pages 1–2, October 2010.
- [10] D. Gautam, Z. Swiecki, D. W. Shaffer, A. C. Graesser, and V. Rus. Modeling classifiers for virtual internships without participant data. In *Proceedings of the 10th International Conference on Educational Data Mining*, pages 278–283, 2017.
- [11] B. Glaser and A. Strauss. *The discovery of grounded theory: Strategies for qualitative research*. Aldine, Chicago, 1967.
- [12] A. C. Graeser, X. Hu, V. Rus, and Z. Cai. Conversation-based learning and assessment environments. In D. Yan, A. A. Rupp, and P. W. Foltz, editors, *Handbook of Automated Scoring*, pages 383–402, New York, February 2020. Chapman and Hall/CRC.
- [13] G. Li and G. Jiabao. Liu, gang, and jiabao guo. "bidirectional lstm with attention mechanism and convolutional layer for text classification. *Neurocomputing*, 337:325–338, 2019.
- [14] Y. Li, R. Krishnamurthy, S. Raghavan, and S. Vaithyanathan. Regular expression learning for information extraction. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 21–30, October 2008.
- [15] D. Shaffer. *Quantitative Ethnography*. Cathcart Press, Madison, WI, 2017.
- [16] D. W. Shaffer and A. R. Ruis. How we code. In *Advances in Quantitative Ethnography: ICQE Conference Proceedings*, pages 62–77, February 2021.

Towards Including Instructor Features in Student Grade Prediction

Nathan Ong
University of Pittsburgh
nro5@pitt.edu

Jiaye Zhu
University of Pittsburgh
jiz188@pitt.edu

Daniel Mossé
University of Pittsburgh
mosse@pitt.edu

ABSTRACT

Student grade prediction is a popular task for learning analytics, given grades are the traditional form of student performance. However, no matter the learning environment, student background, or domain content, there are things in common across most experiences in learning. In most previous machine learning models, previous grades are considered the strongest prognosis of future performance. Few works consider the breadth of instructor features, despite the evidence that a great instructor could change the course of a student's future. We strive to determine the true impact of an instructor by analyzing student data from an undergraduate program and measuring the importance of instructor-related features in comparison with other feature types that may affect state-of-the-art student grade prediction models. We show that adding extensive instructor-related features improves grade prediction, when using the best supervised learning classifier and regressor.

Keywords

Grade Prediction, Student Performance Prediction, Feature Analysis, Instructor Features, Learning Analytics

1. INTRODUCTION

Student performance prediction is a useful service for multiple educational stakeholders in a university and other educational contexts. For example, it is a frequent feature in learning analytics software, like in early-warning systems [6], curriculum personalization [5], cultivating student study skills [13], characterizing course difficulty [31], and can be incorporated into Intelligent Tutoring Systems [22], Massively Open Online Courses [26], and Learning Management Systems [17]. It makes sense that a student would want to use their predicted grade in future courses for short-term course planning, or if an instructor or advisor would want to predict the grades of their students as an early indication of which students are likely to need more assistance.

N. Ong, J. Zhu, and D. Mosse. Towards including instructor features in student grade prediction. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 239–250, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853063>

Intuitively, it would appear that student history would be sufficient to predict future grades. However, in a classroom environment, the student and their past grades are not the only factors that dictate the student's performance. The specific course for which the grade is being predicted and the particular instructor can also affect the outcome of the student's efforts. It is common knowledge that students have varying strengths and weaknesses that interact with courses and their materials. This is reflected in machine learning (ML) models that attempt to carry out student grade prediction by including course-based features. Furthermore, the content of the course and the student's ability to retain the information and skills they learned can also have an impact on student performance (e.g. [27]). Similarly, it is also known that instructors can have a monumental impact on students. For example, good teachers can improve standardized testing scores in reading and math [20]. When considering a teacher's motivation level, there is a direct link to students' academic achievement [4]. Furthermore, it is a common anecdote for someone to be able to point to a teacher that had greatly affected who they became later in life, usually by reinforcing positive traits or shielding from negative influences (e.g. [9]). Yet, an instructor's impact on a student's performance has not been fully explored, quantitatively and in ML models.

In this paper we take a first step to characterize the feature space that can describe the instructor effect on student grade prediction. We experiment with several ML algorithms, using a dataset with thousands of student data records from a large, public university, and show that adding extensive instructor-related features improves grade prediction. Our evaluation shows that GradientBoost is the best supervised learning classifier and regressor, and we will use it to compare instructor-based features with other feature types.

2. PREVIOUS WORK

Student performance prediction is a popular research subject, given its varied applications and approaches. Some works have approached the problem of binary classification of student performance (e.g., predicting pass/fail), to focus educators' attention to the needy students. However, our focus is on overall prediction of the Grade Point Average (GPA) of the final grade in a course, along with a corresponding 5-class breakdown of the grade¹, to explore the possible effect that instructor features have over a course's

¹The categories and corresponding GPA values can be found in Table 3.

final grade. Many works also attempted to train models that predicted multi-class classification (e.g., categorical/letter grading) or regression (e.g., percentage) [1], but with no attempt to include instructor characteristics.

Recent published work in grade prediction has focused on experimenting with different ML models, using features such as student characteristics, domain content, or other course characteristics. Morsy and Karypis [19] focused on assigning knowledge component vectors for each course, paired with a student’s performance in those courses, to inform a regression model, and attained up to 90% accuracy for some predictions with leeway up to 2 half-grades away². Polyzou and Karypis [25] employed Matrix Factorization, previously found in recommendation systems, and focused on historical student grades as the primary feature category, achieving an average error between 2 to 3 half grades. Widjaja et al. [34] combined a Matrix Factorization model, Factorization Machines (FM), with a Long Short-Term Memory neural network, using multiple course and student features, achieving an error between 1 to 2 half grades. None of these works included instructor-based features. For comparison, we will use an FM [29] model as one of our baselines.

Most research to date in this area focuses on past student performance, student-based features, and more recently on course-based features, like in the examples given above. Polyzou [24] went further in an attempt to enumerate student grade and course-based features, but applied the models only to predicting several binary classifications (e.g. will the student fail the course?). They found that for some classification tasks, some feature types improved the performance, while in other classification tasks, they made things worse.

Given that our focus is to add instructor features, we examined works that included them. Ren et al. [28] used only an instructor’s ID when training a neural collaborative filtering model and achieved an error within two half grades. Zhang et al. [35] had a feature list that included 4 instructor-based features (that were grouped with course-based features), namely “teacher’s seniority, teacher’s age, teacher’s title, [and] teacher’s nation.” They applied a variation on a Convolutional Neural Network, achieving an F1-score of 0.805 for a 5-grade classification task. Hu et al. [14] included three instructor features in their datasets, namely instructor’s rank, tenure status, and average GPA over all of the courses they taught in the dataset; they achieved an average error up to 1 grade level difference with regression. Sweeney et al. [32] used 4 instructor features, classification, rank, tenure status, and a bias term introduced in their model, features which focused on their official positions rather than experience, and achieved an error range of two half grades. Each of these works haphazardly included a few instructor features. Only Sweeney et al. analyzed the instructor features’ effects on the performance of their chosen models, and found that the instructor’s bias feature was the third-most important feature, yet was only one-third as important as the student’s bias feature. Our work makes headway in instructor-based features, and examines the number and importance of such features in more detail.

²We define *half grades* to be when the grade changes by one step (e.g., from A to A⁻ or B⁺ to A⁻), while a *full grade* change is when the letter changes (e.g., from A to B)

3. DATASET

3.1 Program Curriculum

The dataset is taken from a computer science (CS) four-year, undergraduate degree program at the main campus of a large American public university. This university has four independent satellite campuses that may offer similar courses, but do not offer the same curriculum. However, it is common for students to transfer from satellite campuses to the main campus and enroll in the CS degree, and some courses students took at the satellite campuses can be transferred into the main campus’ CS program with approval from the main campus undergraduate program director.

Courses in the program are split into three categories: mandatory, electives, and a capstone. There are 8 mandatory courses that all students who intend to graduate with a CS degree must take (unless students will enter with Advanced Placement/International Baccalaureate CS credit or transfer courses from other campuses). For electives, students pick at least 5 upper-level CS courses that pertain to their interests and strengths. Each course may or may not contain prerequisites, and if they do, they tend to be other CS courses. Some mandatory courses have co-requisites, meaning certain courses may be taken at the same time. Students must pass each course with a grade of “C” or higher; if a student does not reach this threshold, they are given the opportunity to retake the course up to two additional times. The last grade that the student received for a course, regardless if it is higher or lower than any previous attempt, is the final grade recorded for the course. The capstone is a project-oriented course as a culmination of the CS curriculum, but not relevant to this research.

Instructors are given flexibility in how they wish to teach their course, as long as they follow the generalized syllabus that is agreed upon by the area faculty for that course. The syllabus contains a list of topics that instructors are expected to cover, but does not prescribe the depth that the instructor must reach for each topic. Should instructors believe there are additional important topics not covered by the generalized syllabus, they are also free to add them into their course. A specific order is similarly not imposed by the generalized syllabus, but topics tend to build on each other naturally, common in STEM fields (e.g. [15]), which imposes a soft ordering. However, due to individual preferences on how to present concepts to students, instructors have the freedom to conduct their courses differently. Given different degrees are offered on satellite campuses, we expect that in the same course, satellite instructors will present their concepts differently from instructors in the main campus.

The main campus CS degree program also is involved in a college-in-high-school (CHS) program, where the CHS program director and a faculty liaison provide materials to high school teachers, and if the student earns a passing grade given by the high school teacher, they qualify for college credit, as if the student took the course at the university. The material high school teachers receive is more structured than the generalized syllabus that instructors for undergraduate students receive, and the university provides training for those teachers to ensure the material that is taught is at the same level of rigor as what is expected for the undergraduate course. The high school teachers assign the final grade,

Table 1: Dataset Statistics

Category	Sub-Category	Count
Grade Records	Raw	186,316
	Non-CE Students	165,655
	Non-CE CS-Course Only	30,672
	Fully Cleaned	28,150
Students	Raw	3,646
	CE Students	524
Courses	Raw	77,733
	CS	4,994
	CS at Main Campus	4,560
	Unique CS	136
	Unique CS at Main Campus	84
Instructors	Raw	12,264 ^{ab}
	All CS	667 ^b
	University Only CS	259
	Main Campus CS	233

^a Upper bound due to 3,786 courses missing instructor data.

^b Upper bound due to 408 CS CHS courses missing instructor data.

Table 2: Student Demographics

Category	Sub-Category	Count	%
Gender	Male	3,013	82.6
	Female	622	17.1
	Other/Unknown	11	0.3
Ethnic Group	American Indian/ Alaskan Native	2	0.1
	Asian	618	17.0
	Black/ African American	131	3.6
	Hispanic/Latino	106	2.9
	Multi-Racial	118	3.2
	White	2,568	70.4
First Generation Undergraduate	Other/Unknown	103	2.8
	First Generation	416	11.4
	Not First Generation	2,424	66.4
Origin	Unknown	806	22.1
	In-State	2,778	76.2
Total	Out-of-State	868	23.8
		3,646	100.0

which are recorded in the official university transcripts.

3.2 Summary Statistics

Transcripts of 3,646 students, all of whom enrolled in at least one of the first two computer science major mandatory courses (since those two can be taken in any order and have no prerequisites) were retrieved from the university registrar. Records spanned between August, 2006 and December, 2019, for a total of 186,316 grade records. Not all students have completed the degree program, but the university does not have an official denotation for when students have decided they no longer wish to pursue their studies at the university or are taking a break from their studies.

3.3 Data Cleaning

To retain consistency, students enrolled in the computer engineering (CE) program, which for a time took many

of the same mandatory courses of the CS program, but with a different passing grade requirement, were removed from the set, leaving 3,122 students. We further cleaned the dataset by removing all non-CS courses and non-letter grades (e.g. “Withdraw”, “Satisfactory”, etc.), leaving 28,150 “fully-cleaned” grade records. Basic statistics can be found in Table 1.

4. METHODOLOGY

Our goal is to generate a supervised ML model that can predict a student’s grade for a target CS course in a given semester of their undergraduate career, and use that model to run a comparison between the different feature types (i.e., to figure out which feature types contribute the most to the best prediction), which will include instructor features.

4.1 Features

We include four types of features in our models: Student Characteristics, Student Grade History, Course Characteristics, and Instructor Characteristics, as detailed below. There are 571 features altogether. The first feature that we include with any model we will train is the target course number, to indicate which course the training grade label came from.

4.1.1 Student Characteristics

Student Characteristics are features that describe the student themselves that are not directly related to their courses. We include a student’s ethnic group, gender, math and verbal SAT scores, ACT scores³, high school GPA, whether they are the first in their family to attend college, and whether they are an in-state or out-of-state student. Furthermore, with the anecdotal knowledge that instructors have particular jargon, preferences, and quirks, we add a feature indicating if the student has ever encountered the same instructor for the target course, or if the instructor was the same for the target course’s prerequisite, in the belief that a student who encounters the same instructor again has a better understanding of how to satisfy the instructor’s requirements. This provides a total of 9 features.

Demographic breakdown of the students in the dataset can be found in Table 2.

4.1.2 Student Grade History

Student Grade History features are a simple enumeration of all CS courses a student has taken in their undergraduate career. Each course is represented by a pair of features, the grade they received as a GPA value (e.g., 4.0 instead of A), and the semester number they took the course relative to the first CS course they took, which includes the summer term. As an example, assume a student took their first ever CS course, CS 101, in the spring term and received a B⁺, and took CS 102 in the fall term of the next school year and received a A⁻; the resulting grade and semester pairs for CS 101 and CS 102 would be (3.25, 0) and (3.75, 2), respectively. We use relative semester value given that students

³The SAT and ACT are common standardized exams that high school students take for entry to an American undergraduate program. For this university, these scores are not required, and are given 0 if no score is provided. Note that 0 is not a valid score for either exam.

have other general education requirements to fulfill not related to the major, and thus can choose to delay taking CS courses, or take the initial CS courses more leisurely. This provides a total of 218 features. Note that each feature will be considered independently when training the models.

The grading scale, GPA-equivalent, and distribution of grade records can be found in Table 3.

4.1.3 Course Characteristics

Course Characteristics describe the courses. We generate these characteristics for the target course, which will be the direct context that the model can use in the grade prediction task. We include the target course’s semester relative to the student’s first semester taking a CS course and maximum enrollment size. We also compile the history of grades for the target course before the semester the student completed the course by providing the parameters that describe their fitted distributions over the GPA-equivalent conversion. We provide the Weibull distribution, which can be described over three parameters: location, shape, and scale, aside from the normal distribution’s mean and standard deviation. Lastly, we derived a feature to denote whether the target class started in the morning (AM) or afternoon and evening (PM). For completeness, we also provide the maximum enrollment size for all courses the student has taken. This provides a total of 117 features.

4.1.4 Instructor Characteristics

Instructor Characteristics describe a target course’s instructor, and were chosen here as an attempt to reflect the instructor’s tendencies and experience, which, under our hypothesis, may have an effect on student performance. We generated the cumulative number of students taught by that instructor in any course to characterize the instructor’s experience. In addition, we include the instructor’s official rank at the time of the target course. In understanding an instructor’s grading behavior, we consider the instructor’s past history of assigning grades, in an attempt to capture how “demanding” the instructor is, by including features that describe the distribution of the instructor’s given grades. We generated the Weibull distribution parameters, location, shape, and scale, along with the mean and standard deviation, for the collection of grades that the instructor has assigned for the target course, for all courses they have ever taught, and for grades they have given for only morning or evening classes. For completeness, we include the instructor’s ID as a base feature for every course the student has taken (or null if the student has not taken the course at the time of the target semester), representing the history of instructors that the student has encountered. This provides a total of 226 features.

Relating to instructor’s official rank, the CS department under study has several official teaching positions, namely “Teaching Fellow,” “Part-Time Instructor,” “Lecturer,” “Senior Lecturer,” “Visiting Lecturer,” “Visiting Professor,” “Assistant Professor,” “Associate Professor,” and “Full Professor.” We opted to merge “Teaching Fellow” with “Part-Time Instructor,” given their duties are exactly the same, but the former corresponds with PhD students. Each instructor’s rank was consistent with the position they held at the beginning of the semester that they taught a course. In addition,

we included another category, “High School Teacher,” to indicate teachers who taught through the college-in-high-school program, given students who are enrolled in the program may have a CHS course (or several) on record. Finally, we include a “Satellite Instructor” title, given the courses taught at the satellite campuses are different, despite the true title that those instructors have.

4.2 Comparison Metrics

We compare the models using weighted average F1-score (Weighted F1) for classification, as well as mean absolute error (MAE) and root mean square error (RMSE) for regression. We also conduct cross-metric comparisons for completeness. For computing MAE and RMSE for each classifier, we converted the predicted letter grade directly into their corresponding GPA value. For computing Weighted F1 for each regressor, we took the predicted value output, which represents an expected GPA, converted it to the closest letter grade, and dropped the plus or minus, where applicable. Conversions between GPA values and letter grades can be found in Table 3.

The F1-score formula is defined as

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} = \frac{\text{TP}}{\text{TP} + \frac{1}{2}(\text{FP} + \text{FN})}$$

where TP is the true-positive rate, FP is the false-positive rate, and FN is the false-negative rate. For multiclass classification, weighted average F1-score formula is defined as

$$\text{Weighted } F_1 = \frac{\sum_c n_c \cdot F_1(c)}{\sum_c n_c}$$

where $F_1(c)$ is the F1-score for class c , and n_c is the number of data points that are part of class c . This metric gives more weight towards correctly predicting larger-sized classes.

The MAE is defined by the following formula

$$\text{MAE} = \frac{\sum_{i=1}^n |y_i - x_i|}{n}$$

where y_i is the value predicted by the algorithm, x_i is the true value, and n is the number of data points. MAE can be preferred for grade prediction because the absolute distance can translate directly into GPA values without additional penalty for significant wrong predictions.

The RMSE is defined by the following formula

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^n (y_i - x_i)^2}{n}}$$

where y_i , x_i , and n have the same definitions as those in the MAE formula. As opposed to MAE, RMSE penalizes large errors, which can be useful in spotting cases where excellent grades are predicted as failing grades, and vice versa.

Note that higher values of Weighted F1 mean the model is better, while for MAE and RMSE, lower values mean that the model is better.

4.3 Data Preparation and Model Selection

Recall that the task is to predict the grade of a student for a target course. Towards that aim, we first clean the dataset

Table 3: Grade class, university’s letter grade, GPA equivalent, and final grade percentage required for the letter grade, along with the percentage of grade records that were given the letter grade and percentage of records for the combined classes.

Class	A		B			C			D			F
Letter Grade	A ⁺ /A	A ⁻	B ⁺	B	B ⁻	C ⁺	C	C ⁻	D ⁺	D	D ⁻	F
GPA	4.0	3.75	3.25	3.0	2.75	2.25	2.0	1.75	1.25	1.0	0.75	0.0
Grade Threshold (%)	92.5	90.0	87.5	82.5	80.0	77.5	72.5	70.0	67.5	62.5	60.0	0.0
Grade Records (%)	28.9	8.7	9.4	18.2	5.7	5.7	11.4	1.0	1.2	3.1	0.5	6.3
Class Records (%)	37.6		33.3			18.1			4.8			6.3

as described in Section 3.3, and then compile the features for each student, ensuring that the feature vector is consistent with the known information at the time they would be taking the course, so that future information cannot inform past courses. For example, if the student is taking CS 102 in the third semester, then only grades from before the third semester will be part of the input. Note that, for the purposes of training the algorithms, we ignore prediction for CS courses offered for non-majors, given that CS students are unlikely to be taking those courses, especially after they have already started taking the mandatory courses. As a result, the total dataset transformed into 25,354 rows, an average of 7.7 transcript moments (or courses and grades) per student considered. The dataset was randomly split 80-20 for training and testing, respectively.

We decide to compare several classifiers and regressors, as described below. We use implementations by Scikit-learn [23], unless otherwise noted. For the classification task, we only require the models to predict from the five letter grades (i.e. {A,B,C,D,F}, where +/- is dropped). For the regression task, we train the models to predict GPA values using the university’s scale (found in Table 3).

We train five different classifiers: majority classification (Majority) as a baseline, decision tree model (DT), K -neighbors model ($K = 2, 2\text{Neigh}$), AdaBoost classifier (AdaClass), and GradientBoost classifier (GradClass). To reduce variability, we utilize five-fold training; we tested typical cross-validation, “soft” voting, and “hard” voting [21]. We explain the first two here for completeness, and we report soft voting because it yielded the best results. In typical 5-fold cross-validation (we assume 5 folds in this paper), five mutually-exclusive and equally large portions of the training set are generated, the classifier is trained on four folds of the data and validated on the fifth, generating a model instance. This procedure is repeated five times, generating 5 different classifier model instances trained on different subsets of the data. We then select the instance with the best performance, and use that model to label new input. In five-fold soft voting, rather than selecting one instance, we average across all instances, and the class with the highest average is the label assigned to the input.

Due to the imbalance of the classes (see Table 3 for a breakdown), we attempted class-balancing via upsampling and downsampling (e.g. SMOTE [8]). However, in the final models trained, we do not perform any rebalancing, because while all models improved on the “C,” “D,” and “F” classes, they did not improve enough to offset the loss of performance in the “A” and “B” classes.

We train seven different regressors. Following the lead of previous works that utilize regression models, we opt to use a Matrix Factorization technique, specifically FM, using a Python wrapper [18] of an existing package [30], along with Linear Regression (LinReg). We also include the regression version of the decision tree model, using both mean-squared error (DT(MSE)) and mean-absolute error (DT(MAE)) to determine the best split, and the K -neighbors model ($K = 2, 2\text{Neigh}$). Finally, we also use the AdaBoost regressor (AdaReg), and GradientBoost regressor (GradReg). We train each model five times, and take the average result.

Using the results from Table 4, the GradientBoost classifier performs the best because it provides the highest weighted average F1-score on the testing set, and has a slight edge on MAE and RMSE over the AdaBoost and decision tree classifiers. When selecting a regressor, the F1-score is less representative due to the prediction of continuous GPA values; even though FM performs better on the weighted average F1-score, we select the GradientBoost regressor because it provides the lowest RMSE and ties with FM on MAE.

5. DISCUSSION

5.1 Individual Feature Weights

Referring to Figure 1, we see that for classification, “instructor_grade_weibull_loc” has the largest feature weight; this feature describes one of the parameters for the fitted Weibull distribution, namely “location.” Location for the Weibull distribution is analogous to the mean for the normal distribution. In our case, the “instructor_grade_weibull_loc” summarizes the instructor’s grades for the target course across all semesters in the dataset before the target semester. Similarly, for regression, the instructor’s mean grade for the target course instead factors as the most-predictive feature. With grading strategies like curving or partial credit, it is easy to see how the instructor’s (subjective) grading style can affect the final grade. We also notice that the difference in contribution between the top-two features is quite large. The difference between the top contributor in classification (approx. 0.16) and the second contributor (approx. 0.10) greatly exceeds the difference between the remaining consecutive features (<0.02). A similar effect can be seen in regression, where the top contributor has a feature importance that is more than double the second feature. This shows that the grades that an instructor is likely to give in a target course is a strong predictor of what kind of grade the student is likely to achieve in the target course. We examine the distribution type in more detail in Section 5.3.

Six out of the top 10 features are grades that the student achieved in the given course number. In this case, all 6 courses are mandatory (out of 8 total) for CS majors to

Table 4: Comparison between different models. Italics represent the best among only classifiers or regressors, while bold represents best overall.

		Training			Testing		
		Weighted F1	MAE	RMSE	Weighted F1	MAE	RMSE
Classifiers	Majority	0.20	1.07	1.55	0.20	1.07	1.55
	DT	0.55	0.57	1.00	0.48	0.65	1.05
	2Neigh	0.68	<i>0.46</i>	<i>0.94</i>	0.40	0.83	1.24
	AdaClass	0.51	0.63	1.02	0.49	0.66	1.06
	GradClass	0.53	0.60	1.02	0.50	0.63	<i>1.04</i>
Regressors	FM	<i>0.60</i>	0.44	0.64	<i>0.47</i>	<i>0.66</i>	0.93
	LinReg	0.40	0.71	0.93	0.39	0.73	0.98
	DT(MSE)	0.48	0.64	0.87	0.44	0.71	0.96
	DT(MAE)	0.50	0.62	0.95	0.45	0.68	1.01
	2Neigh	0.57	0.47	0.66	0.38	0.86	1.16
	AdaReg	0.27	0.80	1.00	0.27	0.81	1.00
	GradReg	0.44	0.66	0.88	0.44	<i>0.66</i>	0.89

take at the university. The second-most predictive feature for both classification and regression are also for the same mandatory course. It seems intuitive that these courses would provide insight into how students would do in future courses, emphasized by the idea that the curriculum requires students to take these mandatory courses first. However, it is likely that the importance of each course is inflated simply because of the mandatory requirement, and as a result provides the initial information that is necessary to predict courses earlier in a student’s undergraduate career. This type of result may not hold as strongly in a major that does not provide a mandatory course schedule.

High school GPA rounds out the top-three. This is consistent with previous literature that indicates that the overall grades received by an incoming student can predict success at the university level [16], from student retention (e.g. [12]) to higher freshmen grades (e.g. [10]). The combined verbal and math score for the SAT also stands out as one of the top features, but is not as predictive as high school GPA, which consistent with the literature [3]. Typically, college-preparedness features are assumed to only have the most impact upon arrival at the university, which is why the typical benchmark for student success that uses these features tends to be freshmen grades. From our work here, it may suggest that these features are more representative of student success throughout the student’s undergraduate career.

While not as predictive as some of the top features, “target_semester” appears as a top-15 feature for both classification and regression, which may suggest that the timing in which a student takes a particular course in their major in relation with other major courses is correlated to their success. However, it is not clear what the causal link may be; it is reasonable to hypothesize that students who take courses in close succession are likely to do better, but it may also be the case that students who do better are more likely to take courses in quick succession. There is also a likelihood that delays in taking courses due to repeating failed courses may be captured by this feature.

5.2 Feature Type Comparisons

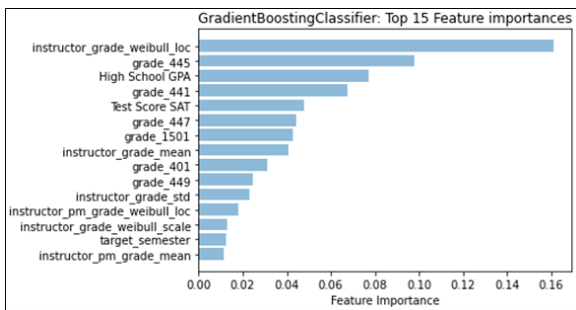
To provide a proper comparison between feature types (each of which is detailed in Section 4.1), we choose to retrain

and retest GradientBoost over each feature type by itself to contrast them with the model trained over all feature types. Given our goal in this paper is to determine the impact that instructor features have on grade prediction models, we also compare a model trained with all feature types except for Instructor Characteristics. We perform the same validation techniques as mentioned in Section 4.3 (5-fold soft vote for classification and 5-run average for regression).

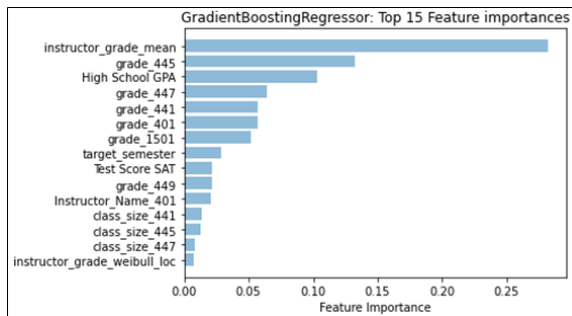
We can see that the Student Grade History feature category continues to be the main feature type for student grade prediction. From Table 5, we note that for all comparison metrics, either the classifier or regressor trained with only Student Grade History features outperforms all other singular feature type trained models. Furthermore, when training GradientBoost over all features, Figure 2 shows that Student Grade History features provides the highest feature weight among all categories. Along with the discussion about course grades from the section above, this provides further evidence to confirm previous research indicating that past student grades are a good predictor for future grades.

Instructor Characteristics comes closely in second, on many of the same angles presented for Student Grade History. From Table 5, we note that the classifier and regressor trained with only Instructor Characteristics has a similar performance with Student Characteristics and outperforms Course Characteristics on all comparison metrics. In terms of feature weights, we also see in Figure 2 that Instructor Characteristics comes closely in second and is almost on par with Student Grade History in the classification task. While individually, an instructor’s grade distribution retains the highest feature weight (as seen in Figure 1), collectively, they still fall short of Student Grade History.

To provide assurance that Instructor Characteristics helps with student grade prediction, we compared two feature sets when training and testing GradientBoost, one trained with all features, and one trained without Instructor Characteristics. Results in Table 5 indicate that for both classification and regression, GradientBoost performs better when Instructor Characteristic features are included. We further examine the classification confusion matrix in Figure 3; in comparing Figures 3a and 3b, we see that adding Instruc-

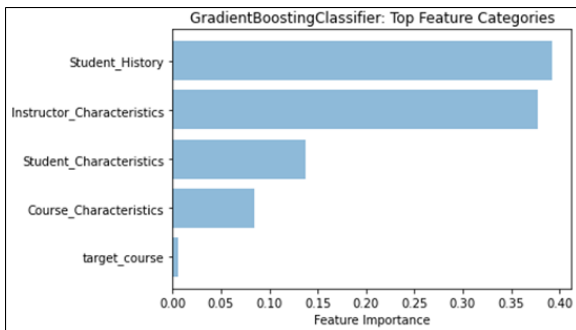


(a) Top 15 features for GradClass

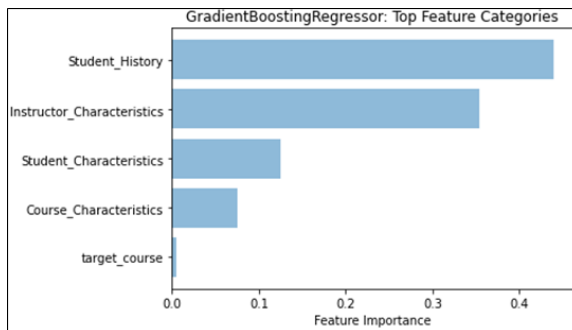


(b) Top 15 features for GradReg

Figure 1: Top 15 features on average utilized by the GradientBoost model when training over all features. Features with names like “grade_(num)” describe the grade that the student received in the indicated course number. Features that start with “instructor_grade” each describes a parameter of the instructor’s grade distribution. “pm” (e.g. in “instructor_pm_grade_mean”) further indicates the grade distribution assigned by the instructor for courses that start in the afternoon or evening.



(a) Top feature categories for GradClass



(b) Top feature categories for GradReg

Figure 2: Top feature categories on average utilized by the GradientBoost model when trained over all features. Feature categories are described in Section 4.1. “target_course” is the feature that describes which course the grade label comes from during training or should be assigned to during testing.

tor Characteristics does have a general positive effect on the accuracy of the classification, given that the number of correct classifications (numbers on the diagonal) increases, and the number of incorrect classifications (numbers not on the diagonal) decreases. Furthermore, we also note that the total misclassifications that are one grade away (adjacent to the diagonal) increases, while the total misclassifications elsewhere decreases. This further confirms that instructor features do have a positive impact in overall classification, not just in singularly increasing recall or precision.

5.3 Grade Distributions and Representation

We tested 103 different distribution types (implemented by SciPy [33]), as well as the logit-normal distribution [2], over the full dataset to determine which kind of distribution best represents the grades’ GPA-equivalent values. We evaluated each distribution using the sum of squared error, and the best fit was the Weibull distribution, which can be represented by three parameters, location, shape, and scale. For every relevant grade distribution, we fit a Weibull distribution to the corresponding grouping, and each parameter was then considered an independent feature. Overall, 10 features were derived from the normal distribution, and 15 were derived from the Weibull distribution. In addition, out of the 25 total features, 5 were Course Characteristics (all grades

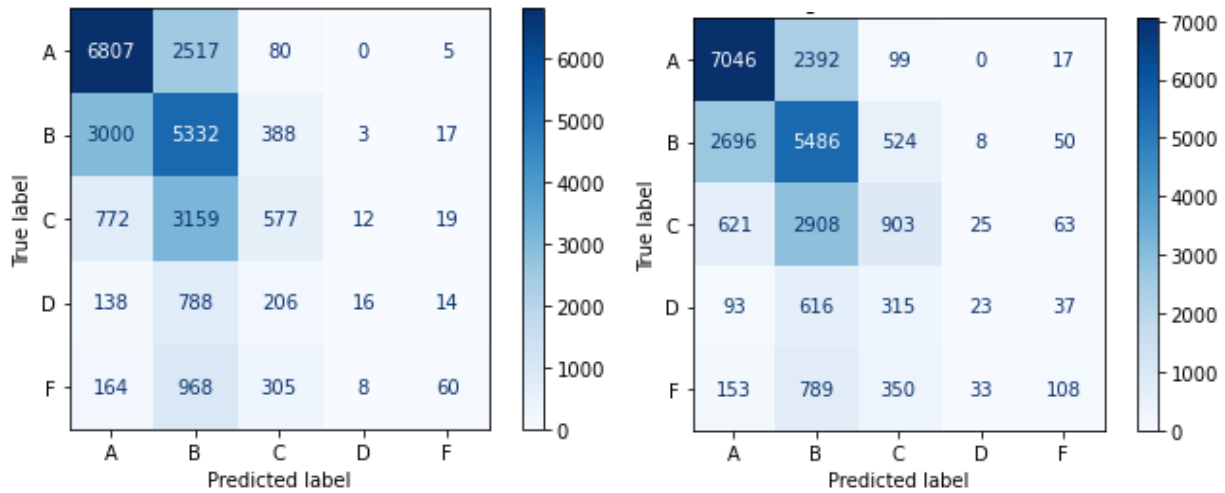
ascribed to the course up to the target semester, regardless of instructor), and 20 were Instructor Characteristics.

There is some controversy about using the normal distribution for representing grades [11], so we briefly investigated the effect of having only the Weibull distribution or the normal distribution represent the grades. We retrained and retested the GradientBoost classifier and regressor with the same procedure in Section 4.3 (5-fold soft vote for classification and 5-run average for regression), both of which are reported in Table 6. From the results, having at least one representation of grade distribution provides some benefit over not having a representation at all, with no difference in performance between the distribution type. Having both provides little-to-no benefit, so it is easy to conclude that it does not matter which grade distribution representation is included, so long as a representation is expressed in the feature set.

Feature weights provided a different angle with which to determine any effects that may stem from the different kinds of distributions. We first examined the effect that historical grades and their distributions had on future grade prediction by examining the weights of those features separately from the main categories. Figure 4 shows that Student Grade History grades and Instructor Category grade distributions

Table 5: Comparison between different feature types. The fifth category combines the first three feature categories together. Italics represent the best among singular feature type, while bold represents best over any category. Note that the “All Features” section mirrors the results from Table 4.

Feature Category	Model	Training			Testing		
		Weighted F1	MAE	RMSE	Weighted F1	MAE	RMSE
Student	GradClass	0.41	0.75	1.18	0.40	0.77	1.18
Characteristics	GradReg	0.30	0.77	1.01	0.29	0.77	1.01
Grade History	GradClass	<i>0.46</i>	<i>0.70</i>	1.13	<i>0.44</i>	0.74	1.16
	GradReg	0.37	<i>0.70</i>	<i>0.93</i>	0.36	<i>0.72</i>	<i>0.94</i>
Course	GradClass	0.39	0.79	1.22	0.36	0.83	1.24
Characteristics	GradReg	0.23	0.80	1.03	0.22	0.81	1.04
Instructor	GradClass	0.42	0.75	1.17	0.38	0.79	1.19
Characteristics	GradReg	0.29	0.78	1.01	0.29	0.79	1.03
Student + Grade + Course Characteristics	GradClass	0.49	0.65	1.06	0.46	0.68	1.09
	GradReg	0.41	0.68	0.91	0.40	0.70	0.94
All Features	GradClass	0.53	0.60	1.02	0.50	0.63	1.03
	GradReg	0.44	0.66	0.88	0.44	0.66	0.89



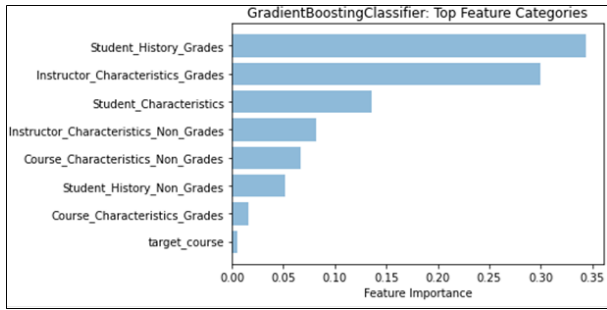
(a) Only Student Characteristics, Student Grade History, and Course Characteristics feature types used in training.

(b) All feature types used in training.

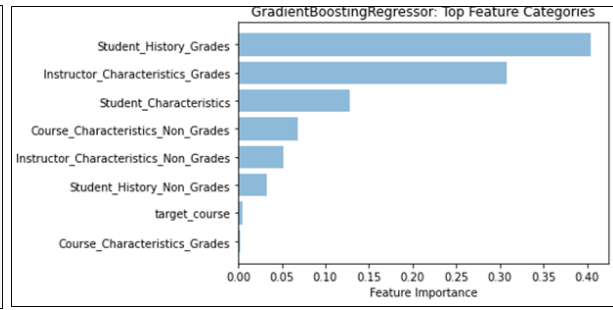
Figure 3: Confusion matrices for the GradientBoost classifier on the test set with the given feature set.

Table 6: Training and testing GradientBoost with different grade distribution types. Note that the rows with all features and both distribution types are the same as those in Table 4.

Feature Set	Distribution Type	Model	Training			Testing		
			Weighted F1	MAE	RMSE	Weighted F1	MAE	RMSE
All Features	None	GradClass	0.50	0.64	1.06	0.46	0.68	1.09
		GradReg	0.40	0.69	0.91	0.39	0.70	0.93
	Weibull Only	GradClass	0.53	0.61	1.02	0.49	0.64	1.05
		GradReg	0.44	0.66	0.88	0.43	0.67	0.90
	Normal Only	GradClass	0.53	0.60	1.01	0.49	0.64	1.05
		GradReg	0.44	0.66	0.88	0.44	0.66	0.89
	Both	GradClass	0.53	0.60	1.02	0.50	0.63	1.04
		GradReg	0.44	0.66	0.88	0.44	0.66	0.89
Instructor Characteristics Only	None	GradClass	0.41	0.79	1.23	0.37	0.82	1.24
		GradReg	0.20	0.80	1.03	0.20	0.81	1.04
	Weibull Only	GradClass	0.42	0.75	1.16	0.39	0.78	1.18
		GradReg	0.29	0.78	1.02	0.29	0.78	1.02
	Normal Only	GradClass	0.41	0.75	1.17	0.39	0.78	1.19
		GradReg	0.29	0.78	1.01	0.29	0.78	1.02
	Both	GradClass	0.42	0.74	1.16	0.39	0.78	1.19
		GradReg	0.30	0.78	1.01	0.29	0.79	1.03

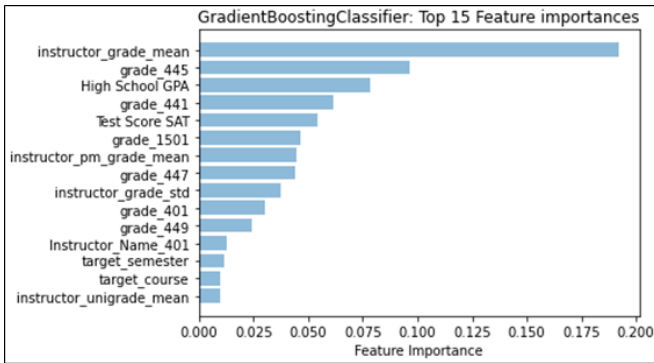


(a) Classification

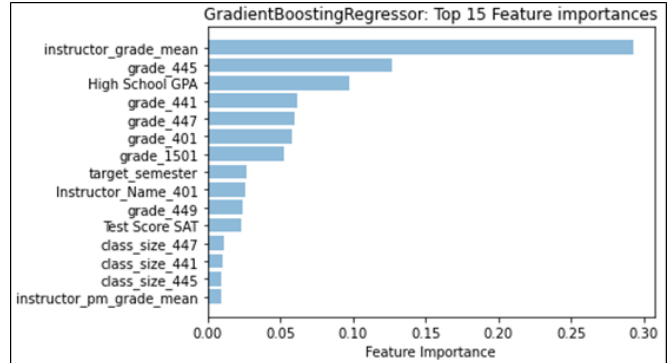


(b) Regression

Figure 4: Top feature categories on average utilized by the GradientBoost model when grades and grade distribution features are separated from their original categories. Feature categories are described in Section 4.1. Categories with the suffix of “Grades” are only features from the category prefix that use GPA as their values, while categories that end with “Non_Grades” are the remaining non-GPA-based features.

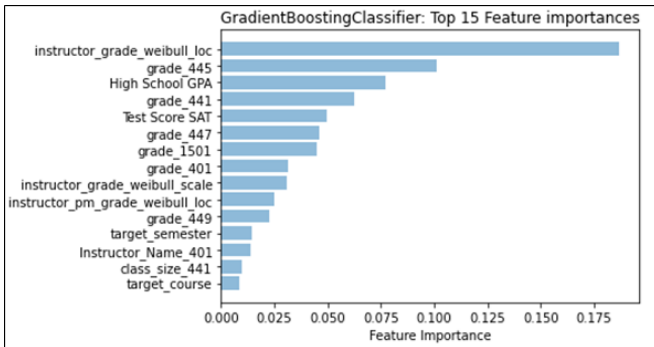


(a) Classification

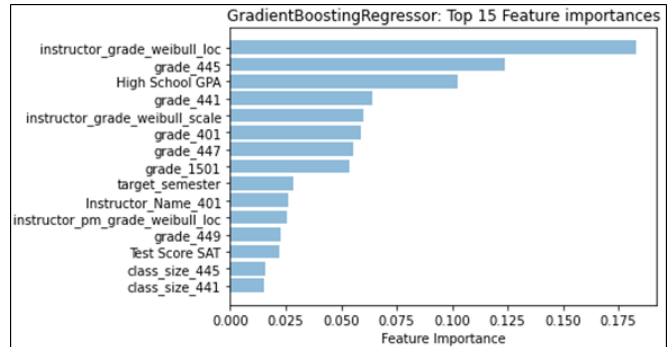


(b) Regression

Normal distribution features only



(c) Classification



(d) Regression

Weibull distribution features only

Figure 5: Top 15 features on average utilized by the GradientBoost model when varying the type of distribution of grades.

would be the top two feature categories, if they were on their own. We then compared the effect that having a singular distribution type has on feature weights, to see if there were any notable changes. Figure 5 shows that no matter which distribution type is being used, the mean or the Weibull distribution’s analogous parameter, location, will remain the top feature overall. However, it is important to note that for the regression task, GradientBoost assigns a much higher weight to the mean than to Weibull’s location parameter,

and even the sum of all three Weibull distribution parameter feature weights, indicating a stronger preference for the normal distribution for regression. We also see this effect appear when both distributions are used, as seen in Figure 1; classification prefers Weibull’s location parameter, while regression prefers the mean. This may provide evidence that the mean captures information that overlaps with more features, more so than the Weibull distribution, given that the performance does not noticeably increase. As for feature

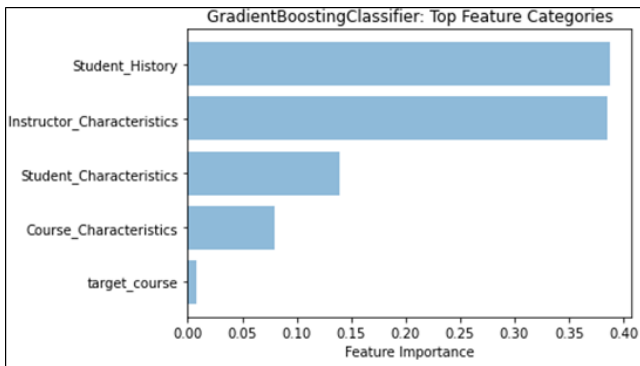


Figure 6: Top feature categories on average utilized by the GradientBoost classifier model when only using the normal distribution to describe grades. Feature categories are described in Section 4.1. “target_course” is the feature that describes which course the grade label comes from during training or should be assigned to during testing.

categories, no placement change was noted. However, in Figure 6, we see that the total feature weight for Instructor Characteristics is almost on par with Student Grade History during classification, further emphasizing the role that the normal distribution can have on predictive models.

5.4 Implications

While our model still has room for improvement for student grade prediction, there are several key items that can be derived from this research.

First, instructor-based features have a place in student grade prediction. The Student Grade History feature type continues to provide the most predictive power for grade prediction, and Instructor Characteristics follows closely behind in second place, while other feature types lag behind.

Second, the distribution of instructor grades is an important feature class to include in future student grade prediction models. It may not matter what kind of grade distribution representation is needed, despite prior research into the “appropriate” distribution, but the parameters for the normal distribution may assist with feature selection, given its high weight in the regression task.

Third, given the strong importance of the distribution of grades that an instructor assigns in their courses, more research is needed to determine the best way to either reduce the impact that instructors have on final grades through teaching ability or subjective measures, or conversely, ensuring that the grades that instructors assigned are truly unbiased and dependent only on the student’s performance in the course. Indeed, it is a long-standing question about the reliability and validity of grades themselves as a measurement of knowledge, given the significant variability in assigning them [7]. One could attempt to expose some of these subjective items by measuring student satisfaction for the instructor, characterizing the instructor’s teaching style, or determining the instructor’s efficacy when utilizing learning management systems, but those would all require additional data collection beyond what a university might readily

have access to.

Lastly, there is still significant room for improvement in explainable student grade prediction. One area where significant work has been done is in Knowledge Tracing to diagnose student issues while they complete a course. Intuitively, understanding how students are doing within a course will ultimately determine how students will do overall, given the knowledge dependency within the course, which is especially important in STEM majors. While Knowledge Tracing does provides additional insight, the effort for training a model significantly increases due to the variation in course content material; it remains to be seen if Knowledge Tracing-adjacent or domain-agnostic Knowledge Tracing features can be generated to assist with generalized student grade prediction without introducing an extra heavy burden.

6. CONCLUSION

In attempting to characterize the relationship between instructor features and predicting a student’s grade, we first enumerated the feature space in grade prediction, with additional emphasis on generating features that describe an instructor’s history and experience in teaching. These features were then extracted from over 13 years and thousands of students’ grade records from a large, public, American university, and used to train and test several supervised ML models. From our experiments, the GradientBoost algorithm, both as classifier and regressor, has the best performance when compared with other supervised ML models.

We then used GradientBoost as our comparison algorithm between different features and feature types, in order to determine the utility of features that define an instructor in a grade prediction model. First, we noted that the distribution of grades that an instructor gives, specifically the mean or the Weibull distribution’s analogous parameter, location, is a major factor in grade prediction. Upon further review, it was found that the distribution representation does not make a major difference in the performance of the model. We then grouped features by type and found that Instructor Characteristics has the second-highest combined feature weight, closely behind Student Grade History. We further trained GradientBoost with and without Instructor Characteristics, and found that Instructor Characteristics contributed to better predictions of student grades for both classification and regression. Therefore, we strongly insist that future ML models should include features that describe Instructor Characteristics, or at the very least, features that describe the distribution of grades that an instructor assigns to a course.

References

- [1] Rahaf Alamri and Basma Alharbi. “Explainable Student Performance Prediction Models: A Systematic Review”. In: *IEEE Access* 9 (2021), pp. 33132–33143.
- [2] Noah Arthurs et al. “Grades Are Not Normal: Improving Exam Score Models Using the Logit-Normal Distribution.” In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*. Montréal, Canada, 2019, pp. 252–257.

- [3] A.W. Astin. *What Matters in College?: Four Critical Years Revisited*. Jossey-Bass higher and adult education series. Wiley, 1997. ISBN: 9781555424923.
- [4] Ken Ayo Azubuike and Orji Friday Oko. “IMPACT OF TEACHERS’ MOTIVATION ON THE ACADEMIC PERFORMANCE OF STUDENTS: IMPLICATIONS FOR SCHOOL ADMINISTRATION”. In: *National Journal of Educational Leadership* 3 (2016), pp. 91–99.
- [5] Michael Backenköhler et al. “Data-Driven Approach Towards a Personalized Curriculum”. In: *Proceedings of the Eleventh International Conference on Educational Data Mining (EDM 2018)*. Buffalo, New York, 2018, pp. 246–251.
- [6] Hall P Beck and William D Davidson. “Establishing an early warning system: Predicting low grades in college students from survey of academic orientations scores”. In: *Research in Higher Education* 42.6 (2001), pp. 709–723.
- [7] Susan M Brookhart et al. “A Century of Grading Research: Meaning and Value in the Most Common Educational Measure”. In: *Review of Educational Research* 86.4 (2016), pp. 803–848.
- [8] Nitesh V Chawla et al. “SMOTE: synthetic minority over-sampling technique”. In: *Journal of artificial intelligence research* 16 (2002), pp. 321–357.
- [9] CNN. *Pushcart classes help break gang chain*. <http://www.cnn.com/2009/LIVING/wayoflife/03/05/heroes.efren.penaflorida/index.html>. Accessed: 2021-09-08. 2009.
- [10] Elchanan Cohn et al. “Determinants of undergraduate GPAs: SAT scores, high-school GPA and high-school rank”. In: *Economics of education review* 23.6 (2004), pp. 577–586.
- [11] Lynn Fendler and Irfan Muzaffar. “The history of the bell curve: Sorting and the idea of normal”. In: *Educational Theory* 58.1 (2008), pp. 63–82.
- [12] Brady P Gaskins. “A ten-year study of the conditional effects on student success in the first year of college”. PhD thesis. Bowling Green State University, 2009.
- [13] Caitlin Holman et al. “Planning for Success: How Students Use a Grade Prediction Tool to Win Their Classes”. In: *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge (LAK 2015)*. Poughkeepsie, New York, 2015, pp. 260–264.
- [14] Qian Hu et al. “Enriching Course-Specific Regression Models with Content Features for Grade Prediction”. In: *Proceedings of the 4th IEEE International Conference on Data Science and Advanced Analytics (DSSA2017)*. Tokyo, Japan, 2017, pp. 462–468.
- [15] Common Core Standards Initiative. *Mathematics standards*. <http://www.corestandards.org/Math/>. Accessed: 2021-09-08. 2021.
- [16] Eunhee Kim et al. “Personal factors impacting college student success: Constructing college learning effectiveness inventory (CLEI)”. In: *College Student Journal* 44.1 (2010), pp. 112–126.
- [17] Andrew E. Krumm et al. “A Learning Management System-Based Early Warning System for Academic Advising in Undergraduate Engineering”. In: *Learning Analytics: From Research to Practice*. Ed. by Johann Ari Larusson and Brandon White. 1st ed. Springer-Verlag, 2014. Chap. 6, pp. 103–119.
- [18] Corey Lynch. *pyFM*. <https://github.com/coreylynch/pyFM>. 2018.
- [19] Sara Morsy and George Karypis. “Cumulative Knowledge-based Regression Models for Next-term Grade Prediction”. In: *Proceedings of the 2017 SIAM International Conference on Data Mining (SDM)*. Houston, Texas, 2017, pp. 552–560.
- [20] Isaac M. Opper. *Teachers Matter: Understanding Teachers’ Impact on Student Achievement*. Santa Monica, CA: RAND Corporation, 2019. DOI: 10.7249/RR4312.
- [21] Behrooz Parhami. “Voting algorithms”. In: *IEEE Transactions on Reliability* 43.4 (1994), pp. 617–629.
- [22] Phil Pavlik Jr. et al. “Using Item-type Performance Covariance to Improve the Skill Model of an Existing Tutor”. In: *Proceedings of the First International Conference on Educational Data Mining (EDM 2008)*. Montréal, Canada, 2008, pp. 77–86.
- [23] F. Pedregosa et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.
- [24] Agoritsa Polyzou. “Models and Algorithms for Performance Prediction and Course Recommendation in Higher Education”. Ph.D. diss. Minneapolis, Minnesota: University of Minnesota, 2020.
- [25] Agoritsa Polyzou and George Karypis. “Grade Prediction with Course and Student Specific Models”. In: *Proceedings of the 20th Pacific Asia Conference on Knowledge Discovery and Data Mining (PAKDD) 2016*. Auckland, New Zealand, 2016, pp. 89–101.
- [26] Zhiyun Ren, Huzefa Rangwala, and Aditya Johri. “Predicting Performance on MOOC Assessments using Multi-Regression Models”. In: *Proceedings of the Ninth International Conference on Educational Data Mining (EDM 2016)*. Raleigh, North Carolina, 2016, pp. 484–489.
- [27] Zhiyun Ren et al. “Grade Prediction Based on Cumulative Knowledge and Co-taken Courses”. In: *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)*. Montréal, Canada, 2019, pp. 158–167.
- [28] Zhiyun Ren et al. “Grade Prediction with Neural Collaborative Filtering”. In: *Proceedings of the 2019 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*. Washington, DC, 2019, pp. 1–10.
- [29] Steffen Rendle. “Factorization Machines”. In: *Proceedings of the 2010 IEEE International Conference on Data Mining (ICDM 2010)*. Sydney, Australia, 2010, pp. 995–1000.
- [30] Steffen Rendle. “Factorization Machines with libFM”. In: *ACM Transactions on Intelligent Systems and Technology* 2.3 (2012), pp. 1–22.

- [31] Sergi Rovira, Eloi Puertas, and Laura Igual. “Data-driven system to predict academic grades and dropout”. In: *PLoS ONE* 12.2 (2017), pp. 1–21.
- [32] Mack Sweeney et al. “Next-Term Student Performance Prediction: A Recommender Systems Approach”. In: *Journal of Educational Data Mining* 8.1 (2016), pp. 22–51.
- [33] Pauli Virtanen et al. “SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python”. In: *Nature Methods* 17 (2020), pp. 261–272. DOI: 10.1038/s41592-019-0686-2.
- [34] Audrey Tedja Widjaja, Lei Wang, and Truong Trong Nghia. “Next-Term Grade Prediction: A Machine Learning Approach”. In: *Proceedings of The 13th International Conference on Educational Data Mining (EDM 2020)*. Fully Virtual, 2020, pp. 700–703.
- [35] Yupei Zhang et al. “Undergraduate Grade Prediction in Chinese Higher Education Using Convolutional Neural Networks”. In: *Proceedings of the 11th International Learning Analytics and Knowledge Conference (LAK 2021)*. Online Everywhere, 2021, pp. 462–468.

Item Response Theory-Based Gaming Detection

Yun Huang¹, Steven Dang², J. Elizabeth Richey³, Michael Asher⁴, Nikki G. Lobczowski³, Danielle Chine¹, Elizabeth A. McLaughlin¹, Judith M. Harackiewicz⁴, Vincent Alevan¹ and Kenneth Koedinger¹

¹ Carnegie Mellon University

yunhuanghci@cmu.edu, {dchine, mimim}@andrew.cmu.edu, alevan@cs.cmu.edu, koedinger@cmu.edu

² Lexia Learning

steven.dang@lexialearning.com

³ University of Pittsburgh

jelizabethrichey@gmail.com, ngl13@pitt.edu

⁴ University of Wisconsin-Madison

{mwasher, jmharack}@wisc.edu

ABSTRACT

Gaming the system, a behavior in which learners exploit a system's properties to make progress while avoiding learning, has frequently been shown to be associated with lower learning. However, when we applied a previously validated gaming detector across conditions in experiments with an algebra tutor, the detected gaming was not associated with learning, challenging its construct validity. Our iterative exploratory data analysis suggested that some contextual factors that varied across and within conditions might contribute to this lack of association. We present a latent variable model, *item response theory-based gaming detection* (IRT-GD), that accounts for contextual factors and estimates latent gaming tendencies as the degree of deviation from normative behaviors across contexts. Item response theory models, widely used in knowledge assessment, account for item difficulty in estimating latent student abilities: students are estimated as having higher ability when they can get harder items correct than when they only get easier items correct. Similarly, IRT-GD accounts for contextual factors in estimating latent gaming tendencies: students are estimated as having a higher gaming tendency when they game in less commonly gamed contexts than when they only game in more commonly gamed contexts. IRT-GD outperformed the original detector on three datasets in terms of the association with learning. IRT-GD also more accurately revealed intervention effects on gaming and revealed a correlation between gaming and perceived competence in math. Our approach is not only useful for others wanting to apply a gaming assessment in their context but is also generally applicable in creating robust behavioral measures.

Keywords

Gaming the system, item response theory, behavior modeling

Y. Huang, S. Dang, J. E. Richey, M. Asher, N. G. Lobczowski, D. Chine, E. A. McLaughlin, J. M. Harackiewicz, V. Alevan, and K. Koedinger. Item response theory-based gaming detection. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 251–262, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license. <https://doi.org/10.5281/zenodo.6853093>

1. INTRODUCTION

Assessing students' engagement levels or motivation from their interaction behaviors in digital learning environments is a compelling challenge both practically and theoretically. Practically, valid behavioral assessment of student engagement can drive adaptations that adjust to students' needs, leading to greater learning and motivation; theoretically, valid behavioral assessment of student engagement can be used to better understand when and why interventions or system designs work for enhancing student learning or motivation. One frequently explored behavioral indicator of student engagement is "gaming the system", which is defined as "attempting to succeed in an educational environment by exploiting properties of the system rather than by learning the material and trying to use that knowledge to answer correctly" [6]. Many studies have demonstrated that gaming the system (abbreviated as *gaming* in this paper) is associated with poor learning outcomes in the short term or the long term [2, 5, 12, 30]. Prior research suggests that interventions directly targeting gaming can reduce gaming behaviors [4, 33] and improve learning [4], demonstrating the practical value of gaming detection. Recent work [28] has also shown that the positive effect of learning with an educational game was fully mediated by lower levels of gaming the system, showcasing the theoretical value of gaming detection for understanding how a specific intervention influences learning.

Past research has leveraged two classes of approaches to model gaming behaviors: knowledge engineering where experts develop rational rules that identify gaming behaviors [21, 25, 26] and machine learning where the model designer creates a set of features first and then a supervised learning algorithm is used to select features for predicting human coded gaming labels [6, 32]. Mainly the emphasis has been put on student features [21, 25, 26], such as how students utilize help (e.g., help abuse [1]) and make errors (e.g., systematic guessing [32]). Task or system features have been investigated to a limited extent although they have been found to be important contextual factors for gaming. For example, [8] found that system features explained more variance in gaming behaviors than student characteristics on a year-long log dataset with 22 different lessons of Cognitive Tutor Algebra. In particular, the results showed that gaming was more frequent in lessons that were abstract, ambiguous, and had unclear presentation of the content or task. Another study [22] also found that differences in gaming behaviors were more strongly associated with the learning

environments than with student populations. Some machine-learned models have incorporated one or two task features in the initial set of features [6, 32]. For example, [32] included features related to question types (top-level or follow-through helping questions) and interfaces (multiple-choice or textbox) in the initial feature set, yet they did not mention whether these features remained in the final model. Meanwhile, in knowledge-engineered gaming detectors, task features typically are not (explicitly) considered [21, 25], i.e., rules to identify gaming behaviors are described in a task type-independent way. Such limited consideration of task features may limit the generalizability of gaming detectors to new contexts with task or system design substantially different from that of the original context. In addition, some research has identified cases where detected gaming behaviors were not harmful [6, 12] or were even good learning behaviors [31], further suggesting the necessity to look carefully into the contexts for identifying or interpreting gaming behaviors. However, such unconventional findings (unconventional in the sense that gaming has frequently been shown to be associated with lower learning) still have not received enough attention in the development of gaming detectors.

Obtaining a student-level gaming estimate has been valuable for studying the relation between student attributes and gaming or intervention effects on gaming. Prior work has predominantly used direct aggregation of detected (or observed) gaming by computing the proportion of gamed transactions or the average of predicted probabilities of gaming for each student [6, 22, 27]. However, [13] showed that the observation-level simple average failed to reveal correlations between motivation and gaming (except for one motivational measure), while a simple latent variable model that estimated a latent gaming tendency for each student controlling for the effect of curricular sections on detected gaming yielded strong associations between a range of motivational measures and gaming tendencies. Inspired by this prior work, we identified an overlooked connection between existing behavior modeling paradigms and knowledge modeling paradigms: latent variable models widely used for estimating student abilities or knowledge levels can also be used to obtain more valid student-level behavioral constructs, thanks to their capacity to account for both task and student features in a single framework. One widely used latent variable modeling paradigm for knowledge assessment in psychometrics is *item response theory* (IRT) [14]. IRT models the observed correctness on each item (e.g., problem steps) of each student as a function of item difficulty and student ability. Instead of using the proportion of correctly answered items as the measure of student ability, IRT accounts for item difficulty in estimating latent student abilities. In essence, students are estimated as having higher ability when they can get harder items correct than when they only get easier items correct. IRT models have been further extended to model dynamic student knowledge by considering the temporal aspect [17], and also by decomposing items into knowledge components (e.g., skills, concepts) shared across items [11].

With the increasing demand of learning engineering efforts towards building effective, engaging learning systems, the generalizability, interpretability, and development cost of gaming detectors are becoming increasingly important. Recent studies [23, 24] compared three previously separately validated gaming detectors across multiple systems: a knowledge-engineered model [25], a machine-learned model [7], and a hybrid model [24] that combines both knowledge engineering and machine learning. In particular, the knowledge-engineered model was developed by using cognitive task analysis to elicit knowledge about how experts code students as gaming or not in Cognitive Tutor Algebra [19]. It

consists of 13 patterns of students' systematic guessing and help abuse behaviors. The comparisons [23, 24] focused on predictive performance of expert labels of gaming in held-out test sets in the original data and two new datasets collected from two other learning environments [3, 27]; the comparison also considered the interpretability of models. Results showed that the knowledge-engineered model achieved greater generalizability to new datasets and interpretability than the machine-learned model, and achieved comparable to slightly better generalizability and interpretability than the hybrid model. Although there was initial cost (higher than that of the machine-learned model) in developing the knowledge-engineered model, it could be directly used in new datasets without further cost (since actions that match any of the 13 patterns can be directly labeled as gaming). However, one may need to retrain the machine-learned or hybrid model (that needs a machine-learned model as input), given the much lower (and even unacceptable) predictive performance of the machine-learned model than the knowledge-engineered model on new datasets [23]. Thus, this knowledge-engineered gaming detector [25], which is referred to as KE-GD in this paper, appears to be the best choice (to build on) among the three detectors, considering generalizability, interpretability, and development cost in a new context altogether; it also represents a broad class of behavioral detectors that are built based on rational rules specified by experts. However, predictive performance of expert labels is only one aspect of construct validity; the establishment of construct validity of a gaming detector also requires examining the association between detected gaming and learning. Past studies [23, 24, 25] have not examined the association between detected gaming by KE-GD with learning, while other studies on other detectors have frequently shown that a higher detected gaming level is associated with lower learning [5, 15, 20, 21, 28].

In this work, we propose a latent variable model, *item response theory-based gaming detection* (IRT-GD), that estimates a latent gaming tendency for each student accounting for contextual factors (i.e., task and student features): students are estimated as having a higher gaming tendency when they game in less commonly (or frequently) gamed contexts than when they only game in more commonly (or frequently) gamed contexts. IRT-GD builds on a previously validated knowledge-engineered gaming detector (KE-GD) that focuses on students' action features and the predictiveness of human labels. We started with applying KE-GD on a dataset collected from experimentation with an algebra tutor, and examined the association between detected gaming and learning, an important aspect for the construct validity of gaming measures. Observing the lack of association with learning, we conducted an iterative exploratory data analysis, and found that this lack of association might result from some contextual factors not considered in KE-GD that varied across and within conditions. Without complex human feature engineering, we integrated contextual factors as predictors in a mixed effect model predicting whether a transaction was detected as gaming by KE-GD, and extracted the student random intercepts as the latent gaming tendencies. We compared KE-GD and IRT-GD by the association with learning in nine contexts, obtained from three datasets and three condition configurations per dataset. Finally, we demonstrated two applications of IRT-GD: to study whether there was a difference in the level of gaming between the two conditions from our experimentation with the tutor, and to explore the relation between gaming and motivation. The development and evaluation process of IRT-GD is explained as follows.

2. DEVELOPMENT OF IRT-GD

2.1 The Tutor

We used datasets collected from an algebra intelligent tutoring system for middle and high school students [18]. Students learn about writing algebraic expressions in story problems in various task (problem) formats: writing an expression in a textbox with dynamic scaffolding steps that appear if a student fails in the original question (*text* format); writing expressions in a table where the main question step and scaffolding steps are accessible at any time and are all required (*table* format); explaining a set of expressions extracted from a given equation by choosing the matching textual description from a dropdown menu for each expression (*menu* format); and given an equation, writing a set of expressions that match a given set of textual descriptions (*flipped-menu* format). These tasks also vary in the complexity of the expressions involved (e.g., one or two operators).

The algebra tutor was continuously redesigned and tested in three experiments with different student populations across three years. In each experiment (eight sessions over four weeks), we compared two versions of the tutor corresponding to two conditions differing in task design and sequencing. The control (CT) condition, corresponding to the original tutor, provided a *normal deliberate practice* schedule. Students received *full* tasks representing the full version of the problem requiring filling in all steps (including scaffolding steps) given a cover story. There were three consecutive units: the first unit contained all the table tasks, the second unit contained less complex menu and flipped-menu tasks, and the third unit contained more complex menu and flipped-menu tasks. Steps were labeled with coarser-grained knowledge components (KCs; skills). Students received individualized practice until reaching mastery of all KCs in a unit before moving on to the next unit. Across the three experiments, the design of the control condition remained the same. The experimental (EXP) condition, the *data-tuned adaptive* condition, corresponds to a redesigned tutor with redesign decisions drawn from data mining outcomes of student log data. It provided an *intense deliberate practice* schedule with task design and sequencing based on a refined, larger KC model revealing hidden difficulties (i.e., original KCs were split to differentiate easier and harder use cases). *Focused* tasks were introduced to reduce over-practicing easier KCs and target particularly difficult KCs. Examples of focused tasks include: text format tasks asking for the final expression without the mandatory intermediate steps required in table task; text format tasks that further remove the story and focus on learning algebraic grammar rules; and simpler menu and flipped-menu tasks with equations less complex than the original equations. There were three or more learning units where different task formats or task types (full or focused) were interleaved in each unit. Students received individualized practice until reaching mastery of all KCs in a unit before moving on to the next unit. Across the three experiments, the design of the experimental condition was continuously refined aiming at promoting greater learning. Our prior work has shown that the experimental condition led to better learning outcomes compared to the control condition [18]. Here, we are interested to see whether intense deliberate practice (i.e., the experimental condition) also led to higher behavioral engagement, particularly lower levels of gaming the system, and also whether gaming was linked with motivation. We started our investigation with the first dataset collected from the first experiment explained below.

2.2 A Previously Validated Knowledge-Engineered Gaming Detector Did Not Generalize

We chose a previously validated knowledge-engineered gaming detector, KE-GD, as the starting point for studying students' behavioral engagement when using the algebra tutor. KE-GD contains 13 interpretable patterns modeling systematic guessing and help abuse. For example, one pattern is "the student enters an incorrect answer, enters a similar and incorrect answer in the same part of the problem and then enters another similar answer in the same part of the problem". It is coded as "incorrect \rightarrow [similar answer] [same context] & incorrect \rightarrow [similar answer] & [same context] & attempt", consisting of constituents such as "[similar answer]" (judged by Levenshtein distance), and action types such as "attempt" (correct or incorrect) or "help". If a sequence of transactions (i.e., student-step interactions considering multiple attempts per step) matches any one of the 13 patterns, then all transactions involved are labeled as gaming. Details of the patterns and the validation of KE-GD could be found in [23, 25].

We used KE-GD to label transactions as gaming or not and then examined the construct validity of detected gaming in our dataset. We defined two metrics of construct validity in the current study, both of which evaluate the association between gaming and learning. The primary metric was the correlation between gaming levels and normalized learning gains over students. For each student, we computed a gaming level using the proportion of gamed transactions (referred to as *proportion of detected gaming* or *detected gaming (proportion)*) for KE-GD, or the estimated gaming tendency for IRT-GD (explained in Section 2.3.2); we computed the normalized learning gain using the widely adopted formula, $(posttest - pretest) / (1 - pretest)$. We used Spearman correlation (ρ) because it is less sensitive to outliers than Pearson correlation. As a supplementary metric, we conducted a regression analysis predicting posttest scores controlling for pretest scores and gaming levels over students and examined the coefficient of the variable of gaming levels. We considered negative correlations and coefficient values at a significance level of 0.10 as acceptable construct validity. Prior studies have used significance levels of 0.05 and 0.10 for correlation analyses involving behavior measures [5, 13, 31].

Two observations emerged. First, the detected gaming proportion 18% (last column in Table 1) was much higher than the previously reported proportions (3.5% in [13] and 6.8% in [25]) of the same detector in other math intelligent tutoring systems. Second, there was a lack of association between detected gaming and learning (correlation: $\rho = -.02$, $p = .86$; regression coefficient: $b = 0.07$, $p = .69$), challenging KE-GD's construct validity in our context.

Table 1. Statistics of the Fall 2019 dataset including the proportion of gamed transactions (considering all attempts of all steps) detected by KE-GD.

#stu	#transaction (tx)	#tx of 1st attempts of steps w/ KCs	Avg proportion of gamed tx over stu
129	98,176	32,419	.18 ($SD = .08$)

2.3 Identifying and Integrating Contextual Factors to Improve Construct Validity

Next, we conducted iterative exploratory data analysis on the first dataset to identify contextual factors that might explain the lack of association between detected gaming by KE-GD and learning, and

integrated the contextual factors through latent variable modeling analogous to item response theory modeling, explained as follows.

2.3.1 Identifying the effect of task formats

One notable feature of our dataset compared to other datasets for developing gaming detectors is that it was collected from experimentation with two conditions with substantial differences in task design and sequencing. So, we first conducted a moderation analysis to test whether the condition moderated the relation between detected gaming and learning. We constructed a regression model predicting posttest scores for each student given the pretest scores, the condition indicator, detected gaming proportion and an interaction term between the condition and detected gaming proportion. The interaction was significant ($b=-0.98, p=.007$) and the control condition showed a relation opposite to theoretical prediction: higher proportion of detected gaming was associated with higher posttest scores (Figure 1).

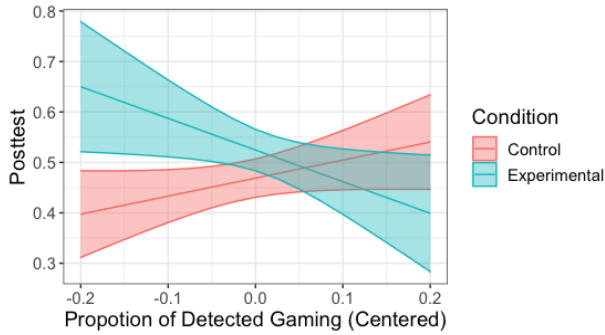


Figure 1. The interaction plot between the condition and detected gaming proportion of the regression model predicting posttest scores with pretest scores controlled for.

To understand this interaction, we started an exploratory data analysis on the overall dataset to examine when and how students gamed according to KE-GD. We used the unit of analysis normally used for modeling student learning, knowledge components (KCs), for better drawing insights into the relation between gaming and learning. We used the KC model previously validated for this dataset [18] based on model fitness. It includes 26 KCs shared by both conditions. We first examined whether students gamed much more on some KCs than on others, and if so whether there was a pattern in this variation. A pattern emerged (see Figure 2) showing that students gamed substantially more on KCs required in menu and flipped-menu formats than those required in table and text formats. Meanwhile, we knew that the control condition positioned menu and flipped-menu tasks in later units, whilst the experimental condition interleaved menu and flipped-menu tasks with text and table tasks in earlier units. Having in mind that higher detected gaming was associated with higher posttest scores in the control condition (Figure 1), we wondered whether this association was because students with higher abilities (who usually also have higher posttest scores) progressed faster to later units and thus accessed a higher proportion of menu and flipped-menu steps, which were highly-gamed contexts, than students with lower abilities. We approximated students' abilities by pretest scores and investigated this relation. Indeed, as shown in Figure 3, students with higher pretest scores in the control condition accessed a higher proportion of menu and flipped-menu steps than students with lower pretest scores (which was not the case for the experimental condition), and as a result, they might appear to game more than students with lower pretest score. Thus, the positive association between detected gaming and posttest scores in the control condition was spurious due to a confounder, the proportion of highly-gamed format steps a

student accessed. The association between detected gaming revealed by KE-GD and posttest scores was biased. If we introduce task formats to account for (part of) the detected gaming, then this bias may be reduced.



Figure 2. Detected gaming proportion by KCs averaged over students. 95% confidence intervals are plotted. (Only first attempts of steps with KCs are considered.)

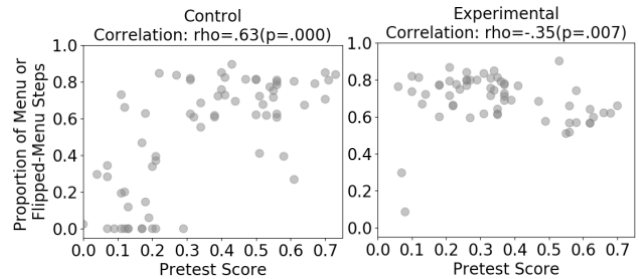


Figure 3. Correlations between pretest scores and proportion of highly-gamed formats (menu, flipped-menu) per condition.

2.3.2 The basic latent variable model accounting for task formats

Based on the first set of exploratory data analyses, we formulated a basic latent variable model, the simplest form of our proposed IRT-GD, that explains detected gaming by both task formats and students' latent gaming tendencies, analogous to explaining item performance by both item difficulties and students' latent abilities in Rasch model [14], the simplest form of item response theory (IRT) models. To illustrate our model, a student with a high proportion of detected gaming due to having a high proportion of menu steps will not be estimated as having a high gaming tendency if he or she does not game more than the average level of the student population on format steps. The model predicts the binary detected gaming label per transaction (i.e., an attempt on a student-step) G asserted by KE-GD, given the student identity and the current format (using a generalized linear mixed model):

$$\text{Detected gaming: } G \sim (1|\text{Student}) + \text{Format} \quad (1)$$

$$\text{Gaming tendency: } \alpha = \exp(\theta) \quad (2)$$

where the student identity is modeled as a random factor and the format of the current step is modeled as a fixed factor. Formula (1)

is written using the syntax of R’s lme4 package for better replicability; a formal mathematical description is that the log odds of a transaction being labeled as gaming by KE-GD is a linear function of the student’s identity (of which the coefficient is the student’s random intercept θ) and the current format. In formula (2), a student’s gaming tendency α is obtained by exponentiating the student’s random intercept θ from formula (1), converting log odds scale to odds scale. This basic model improved over KE-GD in terms of the sign and strength of the association with learning (Table 2 row #1), but the statistical significance was insufficient, demanding further investigation.

2.3.3 Identifying other contextual factors

Based on prior literature, we hypothesized that students’ prior and dynamic knowledge levels (i.e., learning) might also account for detected gaming. The theoretical foundation can be found in several studies: [29] showed that avoiding help and failing repeatedly (which may be considered as systematic guessing, a form of gaming) is associated with better learning than seeking help on steps for which students have low prior knowledge; [31] suggested that the behavior of bypassing abstract hints in search of a concrete solution (traditionally considered as help abuse, a form of gaming) may be an engaged learning behavior where students use bottom-out hints as worked examples; [13] also suggested that detected gaming can be a desirable adaptive learning behavior when students encounter challenges far beyond their abilities. Students’ dynamic knowledge levels were often included as features in machine-learned gaming detectors [6, 32] but absent in KE-GD. Thus, we conducted further exploratory data analysis to examine the effect of prior knowledge (approximated by pretest scores) and dynamic knowledge (approximated by practice opportunities) on detected gaming. More specifically, since we had already identified task formats as an important contextual factor, we hypothesized that there might be interactions between prior knowledge and formats, as well as between practice opportunities and formats, on detected gaming.

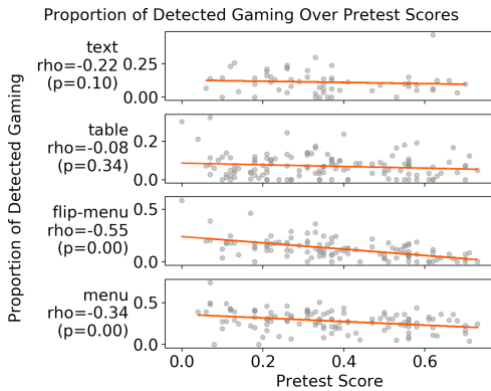


Figure 4. Correlations between pretest scores and detected gaming proportion per task format over students (considering all attempts of all steps).

Figure 4 shows that on flipped-menu and menu formats, students with lower pretest scores gamed much more than students with higher pretest scores, while this was not the case for other formats. Figure 5 shows that on table formats, students were more likely to game on earlier than later opportunities and reduced gaming quickly over opportunities. Discussion of these findings can be found later in Section 5. Based on this second set of exploratory data analysis, we integrated the discovered contextual factors into a latent variable model explained below.

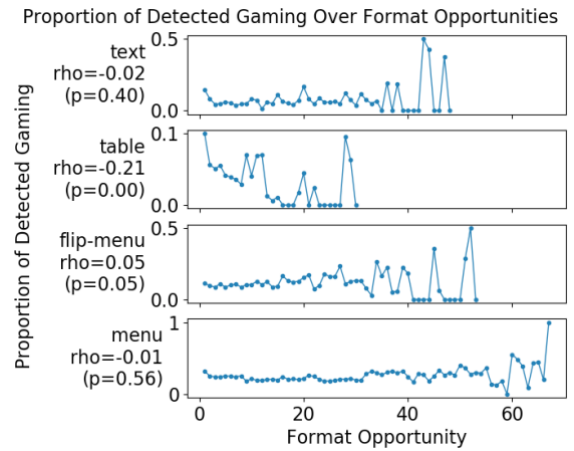


Figure 5. Correlations between practice opportunities and detected gaming per task format. Each point corresponds to the average proportion of detected gaming at an opportunity over students (considering all attempts of all steps). The blips at the end of the curves are due to small sample sizes.

2.3.4 The full latent variable model accounting for critical contextual factors

Based on the second set of exploratory data analyses, we identified two groups of contextual factors that might be important to explain general gaming behaviors: the first group captures the effect of pretest scores adjusted by formats; the second group captures the effect of learning adjusted by formats. We then estimated students’ latent gaming tendencies accounting for these contextual factors. The underlying rationale of our model can also be explained as follows. Since the proportion of detected gaming is usually a low proportion of a full dataset under a study (typically less than 7% in past studies and less than 20% in our datasets), it is sound to assume that a model describing well the detected gaming behaviors of a full dataset captures the *normative* behaviors of a population, and the deviation from the normative behaviors represents the intended gaming construct. Essentially, students are estimated as having a higher gaming tendency when they game in less frequently gamed contexts than when they only game in more frequently gamed contexts according to the general behaviors. This is analogous to IRT models where students are estimated as having higher ability when they can get harder items correct than when they only get easier items correct. The full formulation of our latent variable model IRT-GD is as follows (using a generalized linear mixed model):

$$\begin{aligned} \text{Detected gaming: } G &\sim (1|\text{Student}) + \text{Format} \\ &+ \text{Pretest} + \text{Pretest:Format} \\ &+ \text{Opportunity} + \text{Opportunity:Format} \end{aligned} \quad (3)$$

$$\text{Gaming tendency: } \alpha = \exp(\theta) \quad (4)$$

Formula (3) is written using the syntax of R’s lme4 package for better replicability; a formal mathematical description is that the log odds of a transaction being labeled as gaming by KE-GD is a linear function of the student’s identity (of which the coefficient is the student’s random intercept θ), the format of the current step, the student’s pretest score, the interaction between the pretest score and the format, the practice opportunity count of a format of the student (note that all steps of a task are considered as having the same opportunity count of the corresponding format), and the interaction between the opportunity count and the format. Except for the student identity modeled as a random factor, all other predictors are

modeled as fixed factors. In formula (4), a student's gaming tendency α in odds scale is obtained by exponentiating the student's random intercept θ from formula (3).

Table 2. Associations between gaming tendencies estimated by different variants of IRT-GD and learning. Correlations with normalized learning gains and coefficients of gaming tendency variables in regression predicting posttest scores are reported ($p < .10$: boldfaced and italicized; $p < .05$: boldfaced).

ID	Level	Predictors	Cor with NLG	Post~ Pre+G
1	Format	(1 Stu)+F	$\rho = -.07$ $p = .44$	$b = -.02$ $p = .31$
2	Format	(1 Stu)+F+Pre	$\rho = -.14$ $p = .11$	$b = -0.03$ $p = .16$
3	Format	(1 Stu)+F+Pre+Pre:F ¹	<i>$\rho = -.16$</i> <i>$p = .07$</i>	$b = -0.03$ $p = .14$
4	Format	(1 Stu)+F+Pre+Pre:F+Opp	<i>$\rho = -.16$</i> <i>$p = .07$</i>	$b = -0.03$ $p = .14$
5	Format	(1 Stu)+F+Pre+Pre:F+Opp+F:Opp (The final chosen model)	<i>$\rho = -.18$</i> <i>$p = .04$</i>	<i>$b = -0.04$</i> <i>$p = .09$</i>
6	Format	(1 Stu)+F+Pre+Pre:F+Opp+F:Opp (1st attempts of steps w/ KCs)	<i>$\rho = -.26$</i> <i>$p = .00$</i>	<i>$b = -0.08$</i> <i>$p = .01$</i>
7	KC	(1 Stu)+K+Pre+Pre:K+Opp+K:Opp ² (1st attempts of steps w/ KCs)	<i>$\rho = -.25$</i> <i>$p = .00$</i>	<i>$b = -0.07$</i> <i>$p = .01$</i>

Table 2 shows the construct validity metrics of full models (row #5-#7) as well as reduced models (row #1-#4) of IRT-GD. All the seven variants reached higher validity than KE-GD in terms of having stronger associations with learning, and the three full models reached desirable statistical significance (row #5-#7). The five predictors increasingly strengthened the association (except when adding the single opportunity term in row #4 before adding the interaction term) and were necessary for reaching acceptable validity in this dataset. In formulating the full models, we explored two other configurations: one that used KCs as the unit (row #7) and fit the model using first attempts of steps with KC labels (without modifying the detected gaming labels associated with these transactions); another that used the same data subset as the KC-level model to fit the model but maintaining the unit of format. We found that a format-level modeling worked as well as the KC-level modeling, when using the same subset (row #6 vs. #7). We also found that using the subset with only first attempts of steps labeled with KCs could improve validity compared to using all attempts of all steps (row #6 vs. #5) in this dataset. However, using all attempts of all steps does not require additional KC labels, so we chose to fit IRT-GD with all attempts of all steps for potentially greater generalizability. The final chosen model for the rest of the paper was the one in row #5 in Table 2. We further examined the fitted parameters of the chosen model (Table 3) and found that they had high consistency with the patterns observed in our exploratory data analyses (note that some differences may be due to the differences in statistical methods and data processing used in the two kinds of analyses). We thus concluded the formulation of IRT-GD for valid gaming detection in our tutor.

¹ In R's lme4 package, a colon : is used to denote an interaction term.

² We treated the KC variable as a random factor. R formula:
G~(1|Stu)+(1+Pre+Opp|K)+Pre+Opp.

Table 3. Parameters of the chosen full model of IRT-GD (row #5 in Table 2). Categorical variables were dummy coded and continuous variables were standardized for reducing multicollinearity. The coefficients are in log odds scale.

Modeling purpose	Regression term	Coefficient
Effect of format	Intercept (Text)	$\beta = -2.09$, $p < .001$ ***
	Table	$\beta = -1.50$, $p < .001$ ***
	FlipMenu	$\beta = 0.09$, $p = .03$ *
	Menu	$\beta = 1.12$, $p < .001$ ***
Effect of prior knowledge adjusted by formats	Pretest (Pretest:Text)	$\beta = -0.09$, $p = .13$
	Pretest:Table	$\beta = 0.04$, $p = .37$
	Pretest:FlipMenu	$\beta = -0.19$, $p < .001$ ***
	Pretest:Menu	$\beta = -0.10$, $p = .01$ *
Effect of learning adjusted by formats	Opp (Opp:Text)	$\beta = 0.01$, $p = .71$
	Opp:Table	$\beta = -1.13$, $p < .001$ ***
	Opp:FlipMenu	$\beta = -0.00$, $p = .99$
	Opp:Menu	$\beta = -0.01$, $p = .84$

3. GENERALIZABILITY OF IRT-GD

In the previous section, we conducted exploratory data analysis and validity evaluation on the same dataset and on a single dataset, which might risk overfitting to the dataset. In this section, we tested the generalizability of IRT-GD to two new datasets. We looked into conditions separately and together for all three datasets, resulting in nine contexts across different populations and designs of the system. The two new datasets were collected in 2020 Spring (20S) and 2021 Fall (21F) from the second and third experiments with the tutor with some design changes derived from data mining in the experimental (EXP) condition: new units were introduced for providing focused practice on prerequisite KCs; a lower proportion of menu and flipped-menu tasks was positioned in earlier units compared to the first dataset; a new task format was introduced in the 21F dataset involving interactions with animations. The four task formats identified in the first dataset were still present in the two new datasets. On the other hand, the control condition remained the same.

Table 4 shows statistics of all datasets including detected gaming by KE-GD. Again, the detected gaming proportions were high (16%) in the new datasets. When applying both KE-GD and IRT-GD to the nine contexts (see Table 5), IRT-GD consistently outperformed KE-GD in reaching higher associations with learning in all nine contexts, except one (21F dataset the EXP condition) where the correlation of IRT-GD was slightly weaker but of the same level of significance as KE-GD. In particular, when examining both conditions together and the EXP condition, IRT-GD reached high construct validity (i.e., $\rho < 0$ and $p < .05$) in all six contexts, while KE-GD only reached construct validity in half of the contexts. When examining the control condition, IRT-GD also improved on KE-GD by reversing positive correlations to the theoretically consistent negative correlations for all datasets and reached acceptable significance on the 20S dataset, although the correlations did not reach acceptable significance in other datasets. We conducted further investigation next.

Table 4. Statistics of datasets including detected gaming by KE-GD (CT/EXP: control/experimental condition).

Data	#stu			#transactions	Avg proportion of gamed tx over stu
	All	CT	EXP		
19F	129	69	60	98,176	.18 (<i>SD=.08</i>)
20S	222	106	116	109,193	.16 (<i>SD=.11</i>)
21F	99	46	53	59,703	.16 (<i>SD=.11</i>)

Table 5. Associations between gaming from KE-GD or IRT-GD with learning across nine contexts. The Gaming variables in regression models predicting posttest scores for KE-GD and IRT-GD are of different scales. ($p < .10$: boldfaced and italicized; $p < .05$: boldfaced; NLG: normalized learning gain; CT: control condition; EXP: experimental condition.)

Data	Detector	All		CT		EXP	
		Cor w/ NLG	Post~Pre+G	Cor w/ NLG	Post~Pre+G	Cor w/ NLG	Post~Pre+G
19F	KE-GD	rho=-.02 p=.86	b=0.07 p=.69	rho=.14, p=.25	b=0.34 p=.11	rho=-.29 p=.02	b=-0.71 p=.02
	IRT-GD	rho=-.18 p=.04	b=-0.04 p=.09	rho=-.02, p=.86	b=-0.01 p=.62	rho=-.41 p=.00	b=-0.10 p=.02
20S	KE-GD	rho=-.04 p=.55	b=0.01 p=.92	rho=.16, p=.10	b=0.25 p=.10	rho=-.13 p=.17	b=-0.32 p=.12
	IRT-GD	rho=-.20 p=.00	b=-0.04 p=.00	rho=-.19 p=.05	b=-0.05 p=.03	rho=-.21 p=.02	b=-0.03 p=.06
21F	KE-GD	rho=-.29 p=.00	b=-0.45 p=.00	rho=.00 p=.98	b=-0.04 p=.85	rho=-.52 p=.00	b=-0.83 p=.00
	IRT-GD	rho=-.36 p=.00	b=-0.05 p=.00	rho=-.20 p=.18	b=-.02 p=.19	rho=-.48 p=.00	b=-0.09 p=.00

3.1 Identifying deeper task format effects for refining the input detector

To understand and address the lack of validity of IRT-GD in the control condition in two datasets (Table 5), we conducted further investigation on the 19F dataset where IRT-GD showed the weakest association with learning. We wondered whether the bottleneck lay in the input detector KE-GD. If the gaming labels (i.e., values of the dependent variable for fitting IRT-GD) were too noisy, it would be hard to get accurate tendency estimates by any means. If we decompose a gaming label, it is the union of 13 gaming labels corresponding to 13 gaming patterns defined in KE-GD. Could some of the patterns under some formats be better considered as not gaming in our control condition context? In other words, we hypothesized that there might be deeper task format effects in students' interaction patterns. We conducted a third set of exploratory data analysis where we examined the associations between detected gaming proportions of each of the 13 patterns from KE-GD with learning under each format. We used a *local* normalized learning gain computed using tasks related to a specific format rather than all tasks in the pretest and posttest. The results in Table 6 suggest that on different formats, the same gaming pattern could be helpful or harmful for learning, supporting our hypothesized deeper format effect. To account for this contextual factor, we updated the detected gaming labels from KE-GD in the control condition by using the union of only the patterns that were negatively associated with learning (regardless of statistical significance) for each format while maintaining the labels of the experimental condition. This was a change in the dependent

variable rather than the predictors in IRT-GD. We used the updated dataset to fit new IRT-GD variants, referred to as IRT-GD-PR, and estimated gaming tendencies for the control condition and the overall dataset. Table 7 shows that IRT-GD-PR achieved acceptable validity for the control condition and also boosted the validity for the overall dataset compared to IRT-GD and KE-GD. We leave for future work to further improve and test this local refinement method.

Table 6. Correlations between local normalized learning gains and proportion of each gaming pattern detected by KE-GD in the control condition in the 19F dataset. Pattern #8 was omitted due to its absence. (+: rho>0, -: rho<0, :: p<.10)

Format	1	2	3	4	5	6	7	9	10	11	12	13
Avg prop	.01	.08	.01	.00	.13	.00	.01	.01	.01	.00	.01	.02
Table	-	+	-	-	+	-	-	-	-	+	-	+
Menu	+	-	+	-	-	-	+	-	-	-	-	-
Flip-M	+	-	+	na	-	-	+	+	-	+	-	+

Table 7. Associations between gaming (from KE-GD, IRT-GD, or IRT-GD-PR) with learning. Rho and p values are reported for correlation with normalized learning gains; coefficients and p values of Gaming variables are reported for regression.

Detector	All		CT (control condition)	
	Cor w/ NLG	Post~Pre+G	Cor w/ NLG	Post~Pre+G
KE-GD	-.02(.86)	0.07(.69)	.14(.25)	0.34(.11)
IRT-GD	-.18(.04)	-0.04(.09)	-.02(.86)	-0.01(.62)
IRT-GD-PR	-.27(.00)	-0.07(.01)	-.23(.06)	-0.06(.04)

4. APPLICATIONS OF IRT-GD

In this section, we demonstrated two applications of IRT-GD. We used the estimated gaming tendencies from IRT-GD to study whether there was a difference in the level of gaming between the two conditions from our experimentation with the tutor, and to explore the relation between gaming and motivation.

4.1 Intervention effects on gaming

Our prior work [18] has shown that the data-tuned adaptive condition (that provided intense deliberate practice) led to greater learning outcomes compared to the control condition (that provided normal deliberate practice) in the first experiment (19F dataset); we are interested to see whether the intervention also led to higher behavioral engagement, particularly lower levels of gaming the system. We conducted a regression analysis predicting levels of gaming over students given the condition indicator on the three datasets. The two detectors had contradicting results on the 19F and 20S datasets. On the 19F dataset, KE-GD showed that the intervention led to significantly higher levels of gaming while IRT-GD showed that there was no statistical difference (Table 8 the 2nd column). The suggested intervention effect of increased gaming levels by KE-GD contradicted the previously validated intervention effect of improved learning, since higher levels of gaming are usually associated with lower learning. Thus, IRT-GD more accurately revealed the intervention effect on this dataset. We hypothesized that this could be due to KE-GD not being able to account for the task format effect. We computed the proportion of highly-gamed formats over transactions and the normalized learning gain per student per condition. We found that the EXP condition had a higher average proportion of highly-gamed formats (Table 9 the 2nd column), consistent with our hypothesis. On the

20S dataset, KE-GD showed that the intervention led to significantly lower levels of gaming while IRT-GD showed that there was no statistical difference (Table 8 the 3rd column). However, both conditions have similar normalized learning gains, and the control condition had a much higher average proportion of highly-gamed formats (Table 9 the 20S columns). This again suggests that KE-GD provided biased gaming assessment by using direct proportion of gaming without accounting for formats. This set of analyses shows that IRT-GD more accurately revealed intervention effects on gaming than KE-GD in our experiments.

Table 8. Intervention effects on gaming examined by regression predicting gaming proportions or tendencies given the condition variable (Control: 0, Experimental: 1). Coefficients of the condition variable are reported.

Detector	19F	20S	21F
KE-GD	b=0.02, p=.03	b=-0.09, p<.001	b=0.03, p=.15
IRT-GD	b=-0.05, p=.68	b=-0.03, p=.79	b=0.16, p=.48

Table 9. The proportion of highly-gamed formats (PHGF) in transactions and normalized learning gain per condition. Mean and SD are reported. Higher values are in boldface.

Cond	19F		20S		21F	
	PHGF	NLG	PHGF	NLG	PHGF	NLG
CT	.38(.22)	.16(.29)	.31(.22)	.12(.34)	.16(.17)	.14(.20)
EXP	.44(.09)	.24(.28)	.06(.13)	.14(.39)	.40(.16)	.15(.25)

4.2 Motivation and gaming

The investigation of the relation between motivation and gaming contributes to understanding why students game and developing behavioral measures of motivation. Prior work [9] indicated that students' attitudes and interest towards the domain was related to detected (observed) gaming frequency. More recent work [13] applying a simple latent variable model identified strong associations between several motivational measures and estimated gaming tendencies. Our investigation of the relation between motivation and gaming adds to the limited empirical evidence in this space. On our datasets, motivational surveys with four scales (Table 10) were collected at the first and the last sessions of each month-long experiment. Each question used a 7-point Likert rating; responses for each scale were averaged to present students' motivation along the scale. Table 11 shows correlations between motivational measures from surveys and estimated gaming tendencies over students. Among the four scales, only perceived competence in math (PC) showed consistent significant correlations with gaming and only in the experimental condition across three datasets; the sign of the correlations was negative as theoretically predicted. The correlations between PC and gaming did not appear to be due to students' abilities approximated by pretest scores, because we did not find correlations between pretest scores and gaming tendencies. To understand why PC was only associated with gaming in the experimental condition that provided intense deliberate practice but not in the control condition that provided normal deliberate practice, we compared objective difficulties measured by the proportion correct of first attempts and subjective difficulties measured by the difference between the final and the initial values of PC between the conditions (Table 12). We found that the experimental condition had lower objective difficulties but higher subjective difficulties. We discussed the results in the next section.

Table 10. Motivational survey inventory.

Scale	Question
Perceived competence in math (PC)	How good at math are you?
	Compared to most of your other school subjects, how good are you at math?
Math utility value (UV)	How important is it to you to learn math?
	How important do you think math will be to you in the future?
Interest in math (IM)	How interesting is math to you?
Interest in tutor (IT)	How excited are you to do math on a computer?

Table 11. Correlations between motivational measures from surveys and estimated gaming tendencies. Correlations with pretest scores were added for contrast.

Scale	Cond	19F	20S	21F
PC	CT	-.00(.97)	-.10(.31)	-.03(.84)
	EXP	-.26(.046)	-.18(.05)	-.32(.02)
	All	-.11(.20)	-.15(.03)	-.12(.25)
UV	CT	-.11(.38)	.00(.98)	-.31(.04)
	EXP	.09(.50)	-.03(.78)	-.17(.22)
	All	.01(.94)	-.02(.78)	-.21(.04)
IM	CT	.04(.73)	-.03(.75)	-.05(.73)
	EXP	-.07(.60)	-.13(.18)	-.11(.41)
	All	.02(.87)	-.08(.21)	-.04(.66)
IT	CT	.11(.37)	-.03(.75)	-.01(.97)
	EXP	.06(.64)	-.06(.55)	-.01(.93)
	All	.12(.19)	-.04(.54)	-.01(.93)
Pretest	CT	-.04(.74)	-.01(.91)	-.06(.68)
	EXP	-.01(.92)	-.03(.74)	-.06(.66)
	All	-.01(.90)	-.04(.61)	-.07(.52)

Table 12. Objective difficulties measured by the proportion correct of first attempts (prop cor) and subjective difficulties measured by the difference of PC between the final value and the initial value (Δ PC) per condition. Mean and SD are reported. Higher values are in boldface.

Cond	19F		20S		21F	
	prop cor	Δ PC	prop cor	Δ PC	prop cor	Δ PC
CT	.60(.17)	.02(.86)	.60(.14)	-.03(.83)	.61(.16)	.02(1.11)
EXP	.62(.10)	-.22(.91)	.69(.11)	-.09(.86)	.70(.10)	-.22(.98)

5. DISCUSSION AND CONCLUSION

In this paper, we demonstrate a latent variable model for more valid and robust gaming assessment, item response theory-based gaming detection (IRT-GD), that estimates latent student gaming tendencies accounting for contextual factors. We started with applying a previously validated knowledge-engineered gaming detector (KE-GD) to a dataset collected from an algebra tutor with varying task design and sequencing across conditions. However, the detected gaming level by KE-GD was not associated with learning, challenging its construct validity in our context. We conducted exploratory data analyses and identified contextual

factors that could capture the normative interaction behaviors of the population that might explain this lack of association. We then built an IRT-GD model that explains detected gaming from KE-GD by both contextual factors and students' intrinsic gaming tendencies; it estimates a student-level latent gaming tendency as the degree of deviation from normative behaviors of a population across contexts. We tested the generalizability of IRT-GD and found that it outperformed KE-GD on three datasets across different contexts in construct validity measured by associations with learning. Our approach is not only useful for others wanting to apply a gaming assessment in their context, but is also generally applicable in creating more robust behavioral measures.

There are two notable features of our approach that may be particularly relevant for anyone building or using behavioral detectors. One is that our modeling approach adapts an existing behavioral detector to new contexts without complex feature engineering, which may be attractive for the learning engineering community to maximally build on past methods and adapt them to new contexts. For example, the learning effect on detected gaming is incorporated through practice opportunity counts without an additional process to estimate dynamic knowledge as in [6, 32]. Another feature is that our modeling and evaluation approaches do not require extra human labeling and focus on the association between the behavior measure and learning. Many past works constructed and validated detectors solely by predictions of human labels; although human labels have undeniable merits, they may contain bias. For example, in the development of KE-GD [25], experts examined each clip, which consists of five consecutive actions, from a set of clips randomly selected from log data and decided whether the clip would be coded as gaming or not. A clip was shown in a textual format giving *individual-level* information about the actions *within* the clip (e.g., each action's time, the problem context, the input entered, the relevant skill, whether the input was right, wrong, a help request or a "bug?"), and experts made judgements about gaming without *population-level* information (e.g., the median time of the step of the population), or information *outside* the clip from previous or future clips. This may increase the speed and ease of labeling, yet it may risk introducing bias. For example, if the student did not deviate much from the general behavior of the population or if the student could get a similar step correct in a future clip on their first attempt, then it may be better to label this clip as not gaming. Thus, behavioral detectors validated solely by predictions of human labels looking at isolated clips may not always reliably capture unproductive or harmful behaviors for learning. Our approach reduces bias and enhances the support for learning when applying a behavioral detector by considering contextual factors that were not considered in the original human labeling process, but are important for identifying behaviors harmful for learning. Although further examination of generality, stability, and reliability (as elaborated later) of IRT-GD may be needed to strengthen the validity claim of our approach, we think current evidence suffices to suggest that IRT-GD and our latent variable modeling approach can *enhance* (rather than replace) existing behavior measures for more valid and most robust behavioral assessment. One may consider using IRT-GD and our latent variable modeling approach when an existing behavior measure lacks validity in a specific context.

We identified strong contextual factors, i.e., the task format and its interaction with students' practice opportunities, aligning with previous research. The menu format led to the highest detected gaming, which coheres with prior work hinting at the high propensity of the multiple-choice format (which also involves selecting an option given a set of options) for triggering detected

gaming. One explanation may be that the cognitive cost [16] in making attempts in menus is low since it does not require typing, and different descriptions may only have subtle differences, so students might have developed a trial-and-error strategy with genuine engagement. This explanation could also be applied to the second highest gamed format, flipped-menu, where students might enter several expressions extracted from a given equation corresponding to a description rather than writing expressions from scratch as in other formats. Cognitive cost of a format is implicitly considered in IRT-GD and may be worth more attention for others developing gaming detectors. Meanwhile, students more likely decreased detected gaming as their prior knowledge levels increased on menu and flipped-menu formats compared to other formats, suggesting that certain game-like learning strategy (e.g., a trial-and-error strategy) may only be likely when the cognitive cost is low and students are of low prior knowledge. Moreover, we found that students decreased detected gaming faster on the table format over successive practice opportunities than on other formats, suggesting the reasons for students to game on this format might be different from the reasons they gamed on menus or flipped-menus. Examining the interface, one explanation may be that there are no clear instructions on how to fill in the various cells of the table, e.g., under the column labeled as "Show your work", it is not clear whether a student could enter $15+10$ (graded as wrong) instead of $3*5+10$ (the correct answer). This coheres with prior work suggesting that students gamed more when the presentation is unclear [8], and that students may game as a way to obtain worked examples [31]. Further analysis on student answers may support our hypothesized explanations. A final remark regarding task formats is that in the tutor we studied, the interpretation of task formats requires caution since a task format is not only coupled with a specific interface design (as the name *format* suggests), but also a specific scaffolding design (e.g., fixed or dynamic scaffolding) as well as specific KCs. A future direction is to study them separately through experimentation.

In a context where IRT-GD did not reach statistically significant associations with learning, we conducted local refinement of the input detector, KE-GD, by considering deeper format effects, i.e., the interaction between formats and specific interaction patterns. Our refinement led to acceptable validity and further confirms the importance of task features and demonstrates the flexibility of our latent variable approach. A next step is to test whether this local refinement approach is robust in other contexts. In some contexts, KE-GD already reached acceptable validity, although IRT-GD further improved on it by reaching stronger associations with learning. A next step is to apply IRT-GD to other learning environments and to study more automatic ways to identify contextual factors important in a specific context.

One aspect that needs further examination is whether and how well a fitted IRT-GD model extrapolates to unseen students or formats. This aspect is especially relevant to online intervention where the tutor has to react to gaming as designed for new students or formats. In Section 4, we conducted one kind of generalizability checking where we used the same independent variables in the IRT-GD model for the first dataset to construct IRT-GD models for new datasets fitted to the complete set of the new datasets. The new format (*animation*) was handled through adding a new dummy coded variable. We showed that the structure (i.e., predictors) of IRT-GD generalizes to new students and new versions of the system. This checking is most relevant if one uses IRT-GD to conduct offline student-level analysis as was done in Section 4. However, we have not examined how well a fitted IRT-GD model extrapolates to unseen students or formats, i.e., predicts detected

gaming or estimates gaming tendencies for unseen students or formats, which is important when using IRT-GD for online intervention. In theory a fitted IRT-GD model can extrapolate to an unseen or newly seen student: we first plug in the values of the fixed factors which equates to using the population mean to obtain a prediction; after observing at least one data point of the new student, we can (repeatedly) reestimate the parameters with the accumulated data with a random intercept added for the new student. Meanwhile, the extrapolation to unseen formats is also feasible: we can first treat a new format as a seen, similar format, and after observing at least one data point of the new format, we can (repeatedly) reestimate the parameters with a parameter fitted for the new format³. A promising modification of IRT-GD that enables greater generalizability is to replace the dummy coded format variables with a variable that describes key properties of formats, e.g., *whether a response set is given or can be easily inferred*. As for the question of *how well*, we plan to test the “online” predictiveness of IRT-GD used in the aforementioned ways for extrapolation as a next step.

A related examination that could further support the validity of IRT-GD is to examine the stability and reliability of estimated gaming tendencies. To examine stability, we may check whether gaming tendencies estimated from the first half of students’ temporally ordered interactions correlate with those estimated from the second half of the student interactions, i.e., whether students who tend to game more earlier also tend to game more later. To examine reliability, we may check whether gaming tendencies estimated from interactions of a set of formats correlate with those estimated from interactions of other formats, i.e., whether students who tend to game more in some formats also tend to game more in other formats. The higher the validity and reliability, the higher the truthfulness of the underlying assumption of IRT-GD about a latent, stable gaming tendency construct and the soundness of the identified contextual factors.

Although we have focused on a student-level gaming estimate by IRT-GD, it can also give a transaction-level gaming estimate for online intervention. For example, we can first fit an IRT-GD model to past data using the full formulation. Then, we can apply the fitted model without using the random student intercepts to predict whether an average student may game (as defined by KE-GD) on a step according to current contextual factors. Then we compare this population-level prediction (considering an interval of uncertainty) to the gaming label by KE-GD to identify cases where gaming is not acceptable, i.e., deviating too much from the norm, and finally activate the pre-designed intervention.

One seemingly conflicting result with prior studies is that we did not find an association between gaming tendencies with pretest scores (Table 11), where prior studies have shown that lower prior knowledge levels were associated with higher gaming frequencies [5, 20]. This is because IRT-GD already includes pretest scores and relevant interactions as predictors for detected gaming. IRT-GD is intentionally designed to extract latent gaming tendencies that are not (primarily) triggered by prior knowledge, but by other factors such as students’ motivation or metacognitive skills. This may lead to tutor design that focuses on promoting students’ motivation or metacognition. However, our latent variable modeling approach is flexible in that one could consider dropping the pretest scores

³ Treating a categorical variable with few levels as a random factor may lead to imprecise estimates [10]. Thus, we do not consider this as a next step when the number of formats is small (e.g., <10).

related predictors if they are interested in gaming tendencies triggered by prior knowledge.

One finding seemingly less consistent with prior work and harder to interpret is the link between motivation and gaming. We found a negative correlation between perceived competence in math and gaming in the experimental condition (i.e., the intense deliberate practice condition), consistent with the reported negative correlation between self-efficacy in math and gaming in [13], but we did not find any correlations between other motivational measures and gaming, such as students’ interest towards the domain and gaming reported in [9, 13], or any correlations in the control condition (i.e., the normal deliberate practice condition)⁴. Rather than prematurely attributing the general lack of correlation between motivation and gaming to the lack of validity of estimated gaming tendencies, we hypothesize several reasons. There may be interactions between different student attributes (measured or unmeasured in the current study) or between student attributes and system attributes not considered in a simple zero-order correlation we did here. Additionally, the motivational survey was deployed at the first session but was used to correlate with month-long accumulated behaviors. After all, there is still limited empirical evidence of the relation between motivation and gaming, so further investigation is needed. To explain why there was a negative correlation between perceived competence in math and gaming in the intense deliberate practice condition but not in the normal deliberate practice condition, we conducted a preliminary exploration and found that the objective difficulty (measured by the proportion correct of first attempts) of the intense deliberate practice condition was lower than the normal deliberate practice condition but the subjective difficulty (measured by perceived competence in math) of it was higher. One hypothesis is that the patterns of successes or failures may matter more than the proportion of success for students’ perceived competence. The intense deliberate practice driven by a more fine-grained and larger KC model may have more constantly pushed students to work on their weak spots in new tasks (i.e., put them on the edge of competence), challenging their perceived competence. It may be worth considering letting students to occasionally work on already mastered skills to boost their perceived competence, or preparing students better for desirable difficulties or failures. Combining this finding with the finding that intense deliberate practice alone did not reduce gaming tendencies, one promising direction is to introduce motivational interventions or designs that could maintain or promote perceived competence or self-efficacy in the task domain under intense deliberate practice, to reach a potential multiplier effect of both cognitive and motivational interventions.

6. ACKNOWLEDGMENTS

This work was supported by Bill and Melinda Gates Foundation Prime Award #OPP1196889. We also thank all reviewers for their valuable feedback which helped us improve our manuscript.

7. REFERENCES

- [1] Aleven, V., McLaren, B., Roll, I. and Koedinger, K. 2006. Toward meta-cognitive tutoring: A model of help seeking with a Cognitive Tutor. *International Journal of Artificial Intelligence in Education*, 16(2), pp.101-128.
- [2] Almeda, M.V. and Baker, R.S. 2020. Predicting student participation in STEM careers: The role of affect and

⁴ The positive (rather than the expected negative) correlation rho values in some cells under the 19F column in Table 11 are considered as a result of statistical noise, since none of these values are even marginally significant.

- engagement during middle school. *Journal of Educational Data Mining*, 12(2), 33-47.
- [3] Baker, R.S., Corbett, A.T., and Koedinger, K.R. 2004. Learning to distinguish between representations of data: A cognitive tutor that uses contrasting cases. In *Proceedings of the International Conference of the Learning Sciences* (pp. 58-65).
- [4] Baker, R.S., Corbett, A.T., Koedinger, K.R., Evenson, S., Roll, I., Wagner, A.Z., Naim, M., Raspat, J., Baker, D.J., and Beck, J.E. 2006. June. Adapting to when students game an intelligent tutoring system. In *International conference on intelligent tutoring systems* (pp. 392-401). Springer, Berlin, Heidelberg.
- [5] Baker, R. S., Corbett, A. T., Koedinger, K. R., and Wagner, A. Z. 2004. Off-task behavior in the cognitive tutor classroom: when students "game the system". In *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 383-390).
- [6] Baker, R.S., Corbett, A.T., Roll, I., and Koedinger, K.R. 2008. Developing a generalizable detector of when students game the system. In *User Modeling and User-Adapted Interaction*, 18(3), 287-314.
- [7] Baker, R. S. and de Carvalho, A. M. J. A. 2008. Labeling student behavior faster and more precisely with text replays. In *Proceedings of the 1st International Conference on Educational Data Mining* (pp. 38-47).
- [8] Baker, R. S., de Carvalho, A. M. J. A., Raspat, J., Aleven, V., Corbett, A. T., and Koedinger, K. R. 2009. Educational software features that encourage and discourage "gaming the system". In *Proceedings of the 14th international conference on artificial intelligence in education* (pp. 475-482).
- [9] Baker, R., Walonoski, J., Heffernan, N., Roll, I., Corbett, A., and Koedinger, K. 2008. Why students engage in "gaming the system" behavior in interactive learning environments. *Journal of Interactive Learning Research*, 19(2), 185-224.
- [10] Bolker, Ben. 2022. GLMM FAQ. <https://bbolker.github.io/mixedmodels-misc/glmmFAQ.html#should-i-treat-factor-xxx-as-fixed-or-random>
- [11] Cen, H. 2009. *Generalized Learning Factors Analysis: Improving Cognitive Models with Machine Learning*. Doctoral Thesis. Carnegie Mellon University.
- [12] Cocea, M., Hershkovitz, A., and Baker, R. S. 2009. The impact of off-task and gaming behaviors on learning: immediate or aggregate?. In V. Dimitrova, R. Mizoguchi, B. du Boulay & A. Graesser (Eds.), *Artificial Intelligence in Education* (pp. 507-514). IOS Press.
- [13] Dang, S. and Koedinger, K. 2019. Exploring the Link Between Motivations and Gaming. *Proceedings of The 12th International Conference on Educational Data Mining*, pp. 276 - 281.
- [14] De Boeck, P. and Wilson, M. 2004. *Explanatory item response models: A generalized linear and nonlinear approach*. Springer Science & Business Media.
- [15] Fancsali, S. 2014. Causal discovery with models: behavior, affect, and learning in cognitive tutor algebra. In *Educational Data Mining 2014*.
- [16] Flake, J. K., Barron, K. E., Hulleman, C., McCoach, B. D., and Welsh, M. E. 2015. Measuring cost: The forgotten component of expectancy-value theory. *Contemporary Educational Psychology*, 41, 232-244.
- [17] González-Brenes, J., Huang, Y., and Brusilovsky, P. 2014. General features in knowledge tracing to model multiple subskills, temporal item response theory, and expert knowledge. In *The 7th International Conference on Educational Data Mining* (pp. 84-91).
- [18] Huang, Y., Lobczowski, N.G., Richey, J.E., McLaughlin, E.A., Asher, M.W., Harackiewicz, J.M., Aleven, V., and Koedinger, K.R. 2021. A general multi-method approach to data-driven redesign of tutoring systems. In *LAK21: 11th International Learning Analytics and Knowledge Conference* (pp. 161-172).
- [19] Koedinger, K.R. and Corbett, A. 2006. Cognitive tutors: Technology bringing learning sciences to the classroom. In *The Cambridge handbook of the learning sciences*, 61-77.
- [20] Mogessie, M., Elizabeth Richey, J., McLaren, B.M., Andres-Bray, J.M.L., and Baker, R.S. 2020. Confrustion and gaming while learning with erroneous examples in a decimals game. In *International Conference on Artificial Intelligence in Education* (pp. 208-213). Springer, Cham.
- [21] Muldner, K., Burleson, W., Van de Sande, B., and VanLehn, K. 2011. An analysis of students' gaming behaviors in an intelligent tutoring system: Predictors and impacts. *User Modeling and User-Adapted Interaction*, 21(1), 99-135.
- [22] Paquette, L. and Baker, R. S. 2017. Variations of gaming behaviors across populations of students and across learning environments. *International Conference on Artificial Intelligence in Education* (pp. 274-286). Springer, Cham.
- [23] Paquette, L. and Baker, R. S. 2019. Comparing machine learning to knowledge engineering for student behavior modeling: A case study in gaming the system. *Interactive Learning Environments*, 27(5-6), 585-597.
- [24] Paquette, L., Baker, R.S., de Carvalho, A.M.J.A., and Ocuppaugh, J. 2015. Cross-system transfer of machine learned and knowledge engineered models of gaming the system. In *Proceedings of the 23rd Conference on User Modeling, Adaptation and Personalization* (pp. 183-194).
- [25] Paquette, L., de Carvalho, A. M., and Baker, R. S. 2014. Towards Understanding Expert Coding of Student Disengagement in Online Learning. In *Proceedings of the 36th Annual Cognitive Science Conference*, 1126- 1131.
- [26] Pardos, Z.A., Baker, R.S., San Pedro, M.O.C.Z., Gowda, S.M., and Gowda, S.M. 2014. Affective states and state tests: Investigating how affect and engagement during the school year predict end of year learning outcomes. *Journal of Learning Analytics*, 1(1), 107-128.
- [27] Razzaq, L., Feng, M., Nuzzo-Jones, G., Heffernan, N.T., Koedinger, K.R., Junker, B., Ritter, S., Knight, A., Aniszczyk, C., Choksey, S., and Livak, T. 2005. The ASSISTments project: Blending assessment and assisting. In *Proceedings of the 12th annual conference on artificial intelligence in education* (pp. 555-562).
- [28] Richey, J.E., Zhang, J., Das, R., Andres-Bray, J.M., Scruggs, R., Mogessie, M., Baker, R.S., and McLaren, B.M. 2021. Gaming and Confrustion Explain Learning Advantages for a Math Digital Learning Game. In *International Conference on*

Artificial Intelligence in Education (pp. 342-355). Springer, Cham.

- [29] Roll, I., Baker, R. S. D., Aleven, V., and Koedinger, K. R. 2014. On the benefits of seeking (and avoiding) help in online problem-solving environments. *Journal of the Learning Sciences*, 23(4), 537-560.
- [30] San Pedro, M.O.Z., Baker, R.S.J.d., Bowers, A.J., and Heffernan, N.T. 2013. Predicting college enrollment from student interaction with an intelligent tutoring system in middle school. In *Proceedings of the 6th International Conference on Educational Data Mining*, 177-184.
- [31] Shih, B., Koedinger, K. R., and Scheines, R. 2008. A Response Time Model For Bottom-Out Hints as Worked Examples. *Educational Data Mining 2008*, 117-126.
- [32] Walonoski, J. A. and Heffernan, N. T. 2006a. Detection and Analysis of Off-Task Gaming Behavior in Intelligent Tutoring Systems. *Proc of ITS 2006*, 382-391.
- [33] Walonoski, J.A. and Heffernan, N.T. 2006b. Prevention of off-task gaming behavior in intelligent tutoring systems. In *Proceedings of the 8th International Conference on Intelligent Tutoring Systems* (pp. 722-724), Jhongli, Taiwan. Berlin:Springer-Verlag.

Exploring Cultural Diversity and Collaborative Team Communication through a Dynamical Systems Lens

Mohammad Amin Samadi
University of California Irvine
Irvine, CA
masamadi@uci.edu

Jacqueline G. Cavazos
University of California Irvine
Irvine, CA
jacqueline.cavazos@uci.edu

Yiwen Lin
University of California Irvine
Irvine, CA
yiwenl21@uci.edu

Nia Nixon
University of California Irvine
Irvine, CA
dowelln@uci.edu

ABSTRACT

Collaborative problem solving (CPS) is a 21st-century skill essential for learning gains, workplace success, and tackling increasingly complicated global problems. Group diversity plays a vital role during collaborative activities, especially in a digital space. Although CPS involves dynamic communication behaviors, few studies have considered the impact of cultural diversity on the complex and reoccurring discourse involved in CPS tasks. In this study, we explore team conversations during a CPS task to understand the role of cultural diversity on team communication patterns. First, we characterized team dialogues with an existing CPS framework; then used recurrence quantification analysis (RQA) to quantify group communication and capture recurrent patterns. Finally, we compared the patterns across groups with varying degrees of cultural diversity. Our results suggest that groups with higher levels of cultural diversity, compared to more homogeneous groups, had a higher number of group messages, spent more time in group discussions, and demonstrated greater convergence and complexity in communication patterns. These intricate and complicated communication patterns support the notion that cultural diversity can produce both positive and negative outcomes and may explain the perception of cultural diversity in teams as a “double-edged sword”.

Keywords

cultural diversity, collaborative problem solving, recurrent quantification analysis, group dynamics

1. INTRODUCTION

M. A. Samadi, J. G. Cavazos, Y. Lin, and N. Nixon. Exploring cultural diversity and collaborative team communication through a dynamical systems lens. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 263–275, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853083>

Collaborative problem solving (CPS) involves individuals working together to solve a problem and promotes division of labor, sourcing information from different perspectives and backgrounds, and increasing innovation and creative solutions that stem from the presence of multiple group members [52]. CPS has been identified as a key 21st century skill [9, 80] that plays a pivotal role in both workplace [22] and educational environments [12]. Furthermore, CPS has increasing international importance and has been shown to be an essential skill needed across several domains to solve complex environmental, social, and public health problems [52, 29]. From a socioeconomic perspective, the rise of complicated global issues requires innovative solutions derived from individuals working together and generating solutions from diverse perspectives [22]. In professional environments, employers often report problem-solving [49, 60] and collaboration [49] as skills that are essential for the success and employability of recent college graduates. In educational contexts, CPS is also a key component for successful teamwork and student learning [52, 27, 28]. Positive collaborative interactions have been shown to improve student psychological and performance outcomes [1, 35, 34, 44, 42, 41, 18, 20, 62].

Given the pivotal role of collaboration across multiple disciplines, it is unsurprising that an increasing number of studies have explored which team attributes are important for successful CPS outcomes. For example, studies have considered the impact of group size [68, 45, 57], group diversity [33, 66, 6, 21, 7, 75], and personality differences [37, 32] on team performance. This area of research has consistently highlighted that team diversity in general, and cultural diversity in particular, play a critical role in successful team collaboration [66, 7, 75, 33]. Notably, this line of previous work has generally focused on measuring the effects of team diversity on static post-collaboration measures (e.g., performance outcome) [66, 33, 77]. However, there have been limited research efforts devoted towards understanding the role of diversity from a more dynamic, process-oriented perspective, which are fundamental to collaborative interactions [19, 16, 42]. As such, current studies on diversity in teams are limited in offering insight on many interdependent aspects within collaborative interactions such as negotiation, coordination, and regulation, among others. In order to explore

how these fine-grained collaborative interactions are influenced by diversity, more nuanced techniques are needed.

To address this gap, we use a dynamical systems lens to explore cultural diversity in teams and team communication patterns [58, 15, 79]. Specifically, we apply recurrence quantification analysis (RQA) [78] to quantify team dynamics, and capture recurrent patterns of interaction amongst the members. Using this novel approach, we aim to uncover low-level temporal patterns in CPS communication and variations therein across groups with different levels of cultural diversity. Group communication is inherently interdependent and research has shown that group composition factors are associated with different aspects of social and cognitive processes during collaborative interactions [20, 19, 42, 55, 11, 10, 17]. Therefore, we are motivated to examine whether cultural diversity as a team composition factor is associated with different collaborative communication dynamics and structures.

The remainder of the paper is organized as follows. First, we review the literature on diversity in teams and the impact of diversity on team outcomes and communication. Second, we provide an overview of the current practices in quantifying communication in CPS tasks and the previous work related to RQA in the context of teams and group dynamics. Third, we present our methodological approach including a description of our CPS task, the qualitative coding of CPS skills exhibited in group communication, and the RQA measures. Finally, we present our RQA analysis results with regards to cultural diversity in groups as well as discuss our findings and their implications for understanding communication behaviors in CPS.

2. RELATED WORK

2.1 Diversity in Teams

Diversity in teamwork and its impact on group's performance outcomes has been studied extensively across multiple disciplines and contexts (c.f. [33, 21, 72, 77, 66], for meta-analyses on this topic). However, studies have demonstrated mixed findings on the impacts of team diversity on various team outcomes. Horwitz & Horwitz, 2007 found a positive association between task-related diversity in groups and performance, yet no direct relationship between demographic diversity and group performance. There is some evidence to show that diversity in groups can result in positive outcomes, such as increased innovative and creative ideas [33, 66]. However, other studies suggest that diversity can result in negative outcomes such as increased conflict and lower group cohesion [65, 66]. Although the direct impact of diversity on team outcomes remains unclear [8, 66, 77], several studies have demonstrated that different sources of diversity (personality, cognition, gender, race, ethnicity, etc.) can impact groups to varying degrees [59, 33]. The impact of diversity is often associated with the type of diversity: surface-level diversity (i.e., easily observable attributes such as gender, race, ethnicity) v.s. deep-level diversity (i.e., less overt attributes such as personality, cognition, values/beliefs) [59, 31]. Initially, surface-level attributes may influence group dynamics, but over time deep-level attributes may become more salient and influential [59]. Considering the interactions among the participants in this study and many other collaborations are short-term

(20-minute discussions), surface-level diversity and cultural diversity in particular can prove to be highly influential on group interactions. In the current study, we focus on a surface-level attribute, cultural diversity operationalized by ethnic compositions in group, and its relationship to the communication dynamics during CPS.

2.2 Cultural Diversity in Teams: A Double Edged-Sword

Although there are various forms of group diversity, cultural diversity has often been of particular interest to scientists because of the increased globalized and interconnected workforce [7, 71]. Moreover, an array of collaboration technologies and online platforms enable distance teamwork, and create greater collaborative opportunities for individuals with different cultural backgrounds [22, 29]. Previous studies suggest that the role of cultural diversity on team performance is nontrivial. However, the effect of cultural diversity on group performance is mixed. These effects are often mediated by factors such as creativity, conflicts, communication effectiveness, social integration, and satisfaction [66]. For example, although culturally diverse teams benefit from higher levels of creativity and satisfaction, they also suffer from lower social integration and more conflicts [66]. In addition, it is commonly assumed that people with different cultural and ethnic backgrounds are likely to hold different views and priorities when communicating with others [65, 14]. Following this assumption, higher degrees of cultural diversity in groups may contribute to more complex interpersonal dynamics and information sharing behavior. To extend our understanding of communication patterns associated with cultural diversity, we took on a dynamical system lens to examine structural and temporal processes in CPS discourse.

2.3 Quantifying the Impact of Cultural Diversity

The literature on cultural diversity in teams covers a wide range of domains and collaborative contexts. Existing studies have extensively focused on static team or individual outcomes such as performance [70, 76], number or quality of ideas generated [50, 53], decision-making tasks [81, 40, 64], achieved learning [24], and psychological measures such as learner experience and attitude towards group interactions [54]. These studies have extended our knowledge on the relation between cultural diversity and the end product of the collaboration. However, few studies [69, 73] have considered how cultural diversity impacts more dynamic aspects of the group collaboration process, such as language and group discourse. For example, Tenzer et al., (2014) found that cultural-based language differences can impact group perceptions of trust and competencies. CPS is inherently interactive and dynamic. Collaborative interactions involve reoccurring and interrelated discourse as teams exchange ideas, negotiate, and share information. Thus, it is necessary to unpack collaborative process with regards to cultural diversity compositions. To address this gap, our study focuses on the fine-grained temporal communication patterns across cultural diverse and culturally similar groups.

2.4 Quantifying CPS Communication in Teams

Collaborative problem-solving involves dynamic interpersonal exchange and shared cognitive behavior of individuals [19, 12]. Successful CPS requires multiple skills and subskills for effective communication stages, including negotiation, information sharing, coordination and so on. Language is considered a less intrusive means compared to traditional survey sampling to reveal cognitive processes of the human mind. A number of existing assessment frameworks have been developed to identify and capture CPS skills from interactive dialogues. For example, the Programme for International Student Achievement (PISA) [52], identifies a framework with three social competencies (i.e., establishing and maintaining shared understanding, taking appropriate action to solve the problem, and establishing and maintaining team organization) and four cognitive processes (i.e. exploring and understanding, representing and formulating, planning and executing, and monitoring and reflecting). This framework was first introduced by OECD to evaluate CPS skills during various computer-simulated assessment tasks, highlighting important aspects unique to collaborative interactions mediated by computers. Other CPS frameworks that subsequently surfaced follow similar ontology. For instance, Liu et al. (2016) [43] proposed a framework that conceptualizes four broad CPS skills: sharing ideas, assimilating and accommodating knowledge/perspective taking, regulating problem solving ideas, and maintaining positive communication. Andrews-Todd & Forsyth mapped out more nuanced skills under the social and cognitive domains [2]. More recently, some researchers propose to conceptualize CPS interactions as a continued sociocognitive spectrum, in contrast to the previous dichotomous view on social and cognitive processes [18]. Through this lens, CPS practices could best be described as sociocognitive in nature, allowing more opportunities for adopting computational methods to meaningfully capture natural language patterns of learners. Increasingly, studies have suggested that natural language processing and other artificial intelligence techniques are effective ways to measure CPS skills [13, 74]. Given this trend, researchers have called for more efforts towards developing and adapting innovative data mining methods to effectively quantify CPS patterns and make sense of learner communication behavior [38].

2.5 Communication in Teams as a Dynamical System: RQA

Communication and CPS skills can be considered a dynamic and complex system [46, 23, 36]. Several methods can be used to quantify and analyze group dynamics. However, Knight, et al. (2016) point out that each are limited by a lack of consideration for time dependencies and therefore oversimplification of group dynamics [36]. RQA, on the other hand, is a non-linear dynamical systems approach that enables the evaluation of recurring patterns across times [78]. RQA has been widely used recently to study social interactions, group communication, and group dynamics [3, 4, 36, 25, 26, 15, 67]. For example, Fusaroli & Tylén (2016) took advantage of RQA and cross-recurrence quantification analysis (CRQA) to analyze dyadic conversations patterns and subsequent performance. Based on RQA and CRQA, they defined measures of interactive alignment, interpersonal syn-

ergy, and self-consistency. Interpersonal synergy serve as a significant predictor of performance suggesting the importance of complimentary conversations between the dyads. This dynamic system approach, to our knowledge, has not yet been applied to investigate the communication group dynamics in culturally diverse and homogeneous teams.

In the current study, we use RQA to measure structural components of groups' conversations. Specifically, we focus on time-series data that map CPS skills exhibited in student discourse based on the ontology presented in Andrews-Todd et. al., [2]. RQA results can be visualized using a recurrence plot. The recurrence plot gives a visual representation of the components of each time-series data and plots their recurrence over time. Figure 1 provides an example of RQA in the context of CPS skills. Figure 1A depicts chat discussions during a typical CPS task. Each color in the chat discussion depicts a coded CPS skill (e.g., information sharing) that results in a times series of skills used labeled in Figure 1B. Figure 1C illustrates a recurrence plot based on the chat discussions in Figure 1A. The CPS skills on X and Y axis correspond to the temporal sequence of chat utterances and each dot is an instance in which the CPS skill has recurred in the group conversation. The central diagonal line represents the sequence of CPS skills plotted against themselves, also known as the line of identity (LOI). The recurrence of a skill code used previously in the conversation is represented by dots in the recurrence plot. The recurrence points along with their patterns and alignment provide insight to structural components of group communication. For example, the recurrence of Sharing Information (green) took place at time points four, five, nine, and eleven and the recurrence of Shared Understanding (orange) on times three and eight are recurrence points.

Of particular interest, is any group of dots that create a diagonal line and that are not part of the LOI. A diagonal line of length l is an indication of a recurring pattern of l skill codes in the same sequence. For example, the sequence Monitoring (purple), Shared Understanding (orange), and Information Sharing (green) creates a diagonal line of length 3. These exhibited CPS skills occurs back to back at position, two, three, and four and then again at seven, eight, and nine, thus creating a recurrence of CPS skills.

3. CURRENT WORK

CPS tasks involve dynamic discourse and communication between groups. Therefore, by understanding how cultural diversity impacts these dynamic communication patterns and CPS skills, we can gain insight on the impact of cultural diversity beyond outcome measures such as CPS performance. The current study investigates how cultural diversity in teams impacts group communication behaviors from a dynamical system lens. We explore the linguistic CPS patterns that emerge in culturally homogeneous and diverse groups of undergraduate students as they complete an on-line CPS task. Specifically, we aim to explore the following research questions:

RQ1: Does cultural diversity impact structural components (as captured with RQA) of a group's conversation?

RQ2: How does cultural diversity impact structural compo-

4. METHODS

4.1 Participants

A total of $N = 514$ undergraduate students from a large university in the U.S. southwest participated in the study. Participants were randomly assigned into teams to take part in a Hidden Profile CPS task (described below). In total $N = 129$ teams were included in the study. Teams are predominantly four person groups with a few exceptions that consist of three or five people. In this study we consider four-person groups only for the sake of consistency in analysis. Over a half of our participants were female ($N = 347$), and most of them were freshman ($N = 342$) or junior ($N = 128$). The average age of the participants was 19.5 years, with the 90% of the participants between the ages 18 to 22 years old. Of those participants who reported their race and ethnicity (497 out of 514), 62 (12%) of the participants were White, 9 (1.7%) were Black or African American, 209 (40.6%) were Asian or Asian American, 162 (31.5%) were Hispanic or Latino, and 9 (1.7%) were multiracial. Out of 502 participants who reported their first language the breakdown was as follows: 142 English, 152 non-English, and 208 bilingual. Additionally, over half of students ($N = 277$) identified as first-generation students.

4.2 Procedure

Participants were randomly assigned into teams of four individuals to complete a decision-making task on the Education Platform for Collaborative Assessment and Learning (EPCAL) [30]. EPCAL is a platform by Educational Testing Service that provides a collaboration space for participants to communicate, for teachers or organizers to manage the participants and team formations, and for researchers to study team collaboration in a computer-mediated environment. Prior to beginning the task, participants were asked to complete a background survey to collect information on race, gender, education level, and native language. Next, students were prompted with a problem (e.g., “choose the best apartment”) and were asked to rank three options based on positive features (e.g., “this apartment is at a prime location”) and negative features (e.g., “the rent is expensive”). Teams were randomly assigned to one of four decision-making scenarios including ranking apartments, professors, party venues, and job candidates. Each individual was provided with different features relevant to the problem. In the team discussion phase, participants synchronously chatted with other teammates to share information that they held in order to achieve the optimal ranking. The group communicated through text-based and communication lasted for 20 minutes.

4.3 CPS Skills: Qualitative Coding

In an attempt to qualitatively annotate the utterance data, we adapted the CPS framework from [2]. We removed one cognitive skill code, exploring and understanding, given that it was not as eminent in our data set and was less relevant to our CPS task. The resulting CPS skills are divided into social and cognitive interactions. Each social and cognitive category includes four CPS skill codes, resulting in eight skill codes in total. Social skills include maintaining communication (SMC), sharing information (SSI), establishing shared understanding (SESU), and negotiating (SN); while cognitive skills consist of representing and formulating (CRF),

planning (CP), executing (CE), and monitoring (CM). Table 1 present the definitions and examples of each CPS skill.

CPS skill coding was completed at the chat utterances level and each utterance was assigned one primary code (i.e. eight CPS skill codes aforementioned) and 29 subskills that correspond to each high-level CPS skill. For the purpose of this study, we only focused on the eight main CPS skills. Four undergraduate research assistants were trained as raters to coded the content of students’ discourse (7,711 total utterance events). Raters were trained on the adapted CPS framework. Then, we retrieved a random sample of 20% of all utterances in the data and assigned each rater to code independently. All raters discussed their codes and address any discrepancies. The inter-rater reliability (Kappa = .81) achieved among all raters is considered high (Kappa > .60; [39]). Next, the remaining 80% of the data were split evenly into four groups. One of the four trained raters coded each of these four groups independently.

4.4 Recurrence quantification analysis

We used recurrence quantification analysis (RQA) to visualize and quantitatively assess students’ behaviors within a CPS environment. Specifically, we used PyRQA Python framework¹ [56] to run RQA experiments efficiently. Exploring the recurrence of human behavior would help us better understand how underlying communication patterns occur, and how the phases and dynamics of a system change over time. The time series data required for RQA can be categorical or continuous.

4.4.1 Recurrence plot

Recurrence plot gives us a visual two-dimensional representation to discover repetition, recurrence, and underlying patterns across time from a one-dimensional time series data. For a time-series data t of length ℓ , the recurrence plot R is a $\ell * \ell$ matrix consisting of 0, 1 values, where $R_{i,j} = 1$ is an indication of recurrence between t_i and t_j ($t_i = t_j$). An example of this process is provided in Figure 1.

4.4.2 Recurrence quantification

Visually, the recurrence plot provides valuable qualitative information about the group dynamics and the structure of the dynamical system. However, RQA’s quantified measures, calculated based on the recurrence plot, allow us to quantitatively evaluate a dynamical system beyond visuals and qualitative observations. The following provides a brief description of the main metrics of RQA (for more information on RQA measures see [78, 47]).

- **Recurrence Rate:** Recurrence rate (RR) shows the rate and the density of recurrence points in a recurrence plot. It is calculated by dividing the recurrence points by the total number of cells in the plot which is the length of the time series squared. Higher recurrence rate would show a higher frequency of repetition in actions and the system to revisit previous states. RR ranges from 0 to 1, while 0 shows a system without

¹<https://pypi.org/project/PyRQA/>

Table 1: CPS skills description

	CPS skill code	Definition	Examples
Social	Maintaining Communication (SMC)	Off-Topic Communication, Rapport Building, Inappropriate Communication	<ul style="list-style-type: none"> • “nice job guys” • “no problem”
	Sharing Information (SSI)	Share Own Information, Share Task or Resource Information, Share Understanding	<ul style="list-style-type: none"> • “candidate A was listed as having good leadership skills”
	Establish Shared Understanding (SESU)	Presentation Phase, Acceptance Phase	<ul style="list-style-type: none"> • “What skills do we need?”
	Negotiating (SN)	Express agreement or disagreement, Resolve conflicts	<ul style="list-style-type: none"> • “You’re right” • “My list shows that candidate C is unwilling to further their education.”
Cognitive	Representing and Formulating (CRF)	Represent the problem using words, Proposes specific conceptual thinking	<ul style="list-style-type: none"> • “Yeah I feel that B is the best because everything is nearby and the landlord offers a 24-hour maintenance service”
	Planning (CP)	Set Goals, Develop Strategies	<ul style="list-style-type: none"> • “we have to choose between a and B for being the best”
	Executing (CE)	Suggesting an action to a teammate, Report of own action	<ul style="list-style-type: none"> • “Please list all your features for candidate C”
	Monitoring (CM)	Monitor progress toward the goal, Monitor whether teammates are present	<ul style="list-style-type: none"> • “so we in agreement to make B the best?”

any recurrences, and 1 shows a fully recurrent system.

$$RR = \frac{1}{\ell^2} \sum_{i,j=1}^{\ell} \mathbf{R}(i, j) \quad (1)$$

- **Determinism:** Determinism measures the distribution of recurrence points that form a diagonal line. In other words, determinism is the percentage of recurrence points that align on a diagonal line. The more recurrence points that align on a diagonal line in a recurrence plot, the higher the determinism. As seen in Figure 1, we have four diagonal lines (two diagonal lines of length 3, and two diagonal lines of length 2) we have a fully deterministic system with the determinism of 1. A system with a higher determinism is considered ordered and repetitious with periodic patterns over the time. A system with a lower determinism can be a sign of a more chaotic system. To compute determinism, l_{min} needs to be considered as a minimum length of which diagonal lines to consider. For instance, a $l_{min} = 3$ would eliminate the two diagonal lines of length 2 in Figure 1 and therefore lower the determinism compared to the default value ($l_{min} = 2$).

$$DET = \frac{\sum_{\ell=l_{min}}^N \ell P(\ell)}{\sum_{\ell=1}^N \ell P(\ell)} \quad (2)$$

- **Laminarity:** First introduced by [48], laminarity is designed to capture the percentage of recurrence points that align on a vertical line. A continuation of the same event in a system forms a vertical line. In our study, same skill codes appearing in consecutive messages results in a vertical line. For the computation, a minimum line length of v_{min} needs to be set to possibly restrict the length of vertical lines in the calculation of laminarity.

$$LAM = \frac{\sum_{v=v_{min}}^N v P(v)}{\sum_{v=1}^N v P(v)} \quad (3)$$

- **Entropy of diagonal lines:** Entropy reflects the complexity of the length of diagonal lines in a recurrence plot. It is calculated using the Shannon entropy [63] of $P(\ell)$ distribution, where $P(i)$ is the probability to find a diagonal line of length i in the recurrence plot. High entropy is an indication of high variation in length of recurrent sequences. In contrast, in a recurrence plot where all diagonal lines are the same length, the entropy would be zero.

$$p(\ell) = \frac{P(\ell)}{\sum_{\ell=l_{min}}^N P(\ell)}, \text{ENTR} = - \sum_{\ell=l_{min}}^N p(\ell) \ln p(\ell) \quad (4)$$

Table 2: Descriptive statistics of measures by cultural diversity

Measures	Cultural diversity degree				
	1	2	3	4	Full Sample
	$M(SD)$	$M(SD)$	$M(SD)$	$M(SD)$	$M(SD)$
Determinism	0.52 (0.2)	0.5 (0.15)	0.46 (0.15)	0.56 (0.12)	0.49 (0.15)
Divergence	0.22 (0.1)	0.22 (0.16)	0.24 (0.17)	0.12 (0.06)	0.22 (0.16)
Entropy	0.96 (0.62)	0.92 (0.47)	0.81 (0.34)	1.22 (0.44)	0.9 (0.43)
Laminarity	0.71 (0.15)	0.67 (0.14)	0.66 (0.13)	0.72 (0.11)	0.67 (0.13)
Longest diagonal line	6.29 (4.07)	6.41 (4.28)	5.62 (2.93)	11.24 (6.06)	6.48 (4.21)
Recurrence rate	0.24 (0.06)	0.25 (0.05)	0.25 (0.06)	0.25 (0.06)	0.25 (0.06)
Discussion Time	715.15 (183.38)	727.34 (358.15)	885.42 (431.21)	1141.0 (285.84)	850.47 (404.31)
Average diagonal line	2.9 (1.23)	2.72 (0.73)	2.44 (0.37)	3.2 (1.25)	2.63 (0.74)
Number of messages	39.86 (11.02)	49.96 (31.14)	59.24(31.93)	77.47 (27.38)	57.0 (31.69)

notes: M = Mean, SD = Standard Deviation

- **Longest Diagonal Line and Divergence:** Another measure related to diagonal lines are (1) the longest diagonal line, L_{max} , excluding the line of identity, and (2) divergence which is the reverse of L_{max} . Along with determinism, these two measures are indicators of convergence, chaos, and stability in the system. The lower the L_{max} , the higher the divergence, chaos, and instability within the dynamical system.

$$DIV = \frac{1}{L_{max}} \quad (5)$$

4.5 Data Processing

4.5.1 RQA time series

To prepare the data for RQA, we created time series data using the skill codes associated with each chat within the same conversation. The utterances were grouped together based on the team identification code and therefore each group conversation is represented by a time series of messages and their associated skill codes. Categorical skill codes were mapped to a numeric code that represents that skill throughout the analysis. The mapping is one-to-one meaning that each skill is only mapped to one number and each number only represents one skill. RQA was then applied to the time series of numeric skill codes to explore structural patterns within the group conversation.

4.5.2 Cultural diversity

We quantified cultural diversity based on the heterogeneity of ethnic identities. To determine cultural diversity level, we calculated how many unique ethnicities existed in each group according to the students self-reported demographic information. Since groups consist of four members, there were four possible levels of group diversity ranging from fully homogeneous groups (coded as 1) to fully heterogeneous groups (coded as 4). Below is a breakdown of group compositions and their associated degrees of cultural diversity:

- 1: (4 White members), $N = 7$;
- 2: (3 White members, 1 Asian member) or (2 White members, 2 Asian members), $N = 42$;
- 3: (2 White members, 1 Asian member, 1 Hispanic member), $N = 67$;

- 4: (1 White member, 1 Asian member, 1 Hispanic member, 1 Black member), $N = 13$

4.5.3 Statistical Analysis

RQA was applied to a time series of CPS skill codes associated with each message in a conversation. As described in Section 4.4, RQA allowed us to study team dynamics, underlying behavioral patterns, and their complexity. In order to examine the influence of cultural diversity on CPS communication dynamics, we performed a Kruskal-Wallis test on each RQA measure to see whether structural components of dialogues were different across groups. In addition to seven RQA measures described, we also included the total amount of time (seconds) that students spent discussing their decision and the number of messages sent within that discussion time as conversational measures. We also conducted pairwise Wilcoxon rank tests as a post-hoc analysis with Benjamini-Hochberg p-value adjustment to further locate where the significant difference specifically resides. This analysis enabled us to detect significance of variations of conversational measures and their relation to the level of team diversity. The code and results of the RQA Analysis is available on Github at: github.com/The-Language-and-Learning-Analytics-Lab/cult-div-rqa

5. RESULTS

Descriptive statistics of conversational measures by cultural diversity is available in Table 2. Results for each Kruskal-Wallis tests on Table 3 suggest significant effect of cultural diversity on entropy at [$H(3) = 9.077, p = 0.029, \eta^2 = 0.049$], longest diagonal line at [$H(3) = 14.405, p < 0.01, \eta^2 = 0.091$], average diagonal line at [$H(3) = 10.858, p = 0.01, \eta^2 = 0.063$], longest diagonal line and divergence at [$H(3) = 14.405, p = 0.003, \eta^2 = 0.091$], number of messages sent at [$H(3) = 11.231, p = 0.01, \eta^2 = 0.066$], and the time spent in discussion at [$H(3) = 12.334, p = .007, \eta^2 = 0.075$]. Notably, due to the relationship of divergence and longest diagonal line in Equation 5, the same results for these two measures were expected. Combined, these suggest structural differences in conversations across the four groups, in terms of complexity of activity and patterns of behavior. We followed up this significant result with pairwise Wilcoxon rank tests reported in Table 4.

The post-hoc analysis suggests significant differences between the most diverse group and the rest of the diversity

Table 3: Results of Kruskal-Wallis test of conversational and recurrence measures by diversity

Conversational measure	<i>df</i>	χ^2	<i>p</i>	η^2
Recurrence Rate	3	0.669	0.880	-0.019
Determinism	3	6.354	0.096	0.027
Laminarity	3	3.678	0.300	0.005
Entropy of Diagonal Lines	3	9.077	0.029*	0.049
Average Diagonal Line	3	10.858	0.010**	0.063
Longest Diagonal Line	3	14.405	0.003**	0.091
Divergence	3	14.405	0.003**	0.091
Discussion Time	3	12.334	0.007**	0.075
Number of Messages	3	11.231	0.010**	0.066

notes: χ^2 = Chi-Squared, *df* = degrees of freedom

. = $p < .1$, * = $p < .05$, ** = $p < .01$, *** = $p < .001$

Table 4: Pairwise comparisons using Wilcoxon rank sum exact test presented by *p* values of recurrence and conversational measures grouped by degree of cultural diversity

Entropy			
Diversity	1	2	3
2	0.753	—	—
3	0.753	0.370	—
4	0.120	0.120	0.015*

Divergence			
Diversity	1	2	3
2	0.849	—	—
3	0.852	0.849	—
4	0.070	0.005**	0.001***

Discussion Time			
Diversity	1	2	3
2	0.812	—	—
3	0.433	0.080	—
4	0.010*	0.004**	0.067

Average diagonal line			
Diversity	1	2	3
2	0.681	—	—
3	1.00	0.145	—
4	0.172	0.145	0.010**

Longest diagonal line			
Diversity	1	2	3
2	0.849	—	—
3	0.852	0.849	—
4	0.070	0.005**	0.001***

Number of Messages			
Diversity	1	2	3
2	0.786	—	—
3	0.112	0.112	—
4	0.020*	0.020*	0.071

groups in most occasions, with the most difference apparent in time spent on discussions and longest diagonal line. The most culturally diverse groups spent more time (in seconds) discussing the problem ($M = 1141$, $SD = 285.84$) than all three other culturally demographic compositions (Level 1 diversity, $M = 715.15$, $SD = 183.38$; Level 2 diversity, $M = 727.34$, $SD = 358.15$; Level 3 diversity, $M = 885.42$, $SD = 431.21$). Descriptive statistics of the measures grouped by degrees of cultural diversity are available in Table 2. These findings further demonstrate the impact of cultural diversity on the group dynamics and key components of group discussions. Figure 3 visualizes the linear relationship between the RQA measures explored in the post-hoc analysis and the amount of cultural diversity in each group. Of the five RQA measures examined, we observed a steady increase in discussion time, number of messages, and longest diagonal line as degree of cultural diversity increased. In addition, both entropy and average diagonal line increased at a less consistent rate. Specifically, entropy and average diagonal line increased mostly between the second most culturally diverse groups and the most culturally diverse groups. We

discuss the implications of the results in the next section.

6. DISCUSSION

CPS has been increasingly recognized as an essential 21st century skill in both educational and workplace environments [22, 12], especially in settings where teams are becoming increasingly more diverse and international. As such, exploring ways to further our understanding of the communication behavior patterns in diverse teams and advance CPS skills has been at the forefront of educational research [12, 29]. In this study, we extended current literature that measures performance outcomes by using a dynamical systems lens to examine how the level of in-group diversity influences team communication behavior. Specifically, we quantified CPS skills exhibited in group discourse and characterized conversational structures through RQA measures.

In response to RQ1, we found that degrees of cultural diversity in teams are associated with systematic outcomes of group communication captured through RQA measures. Specifically, these outcomes include number of messages,

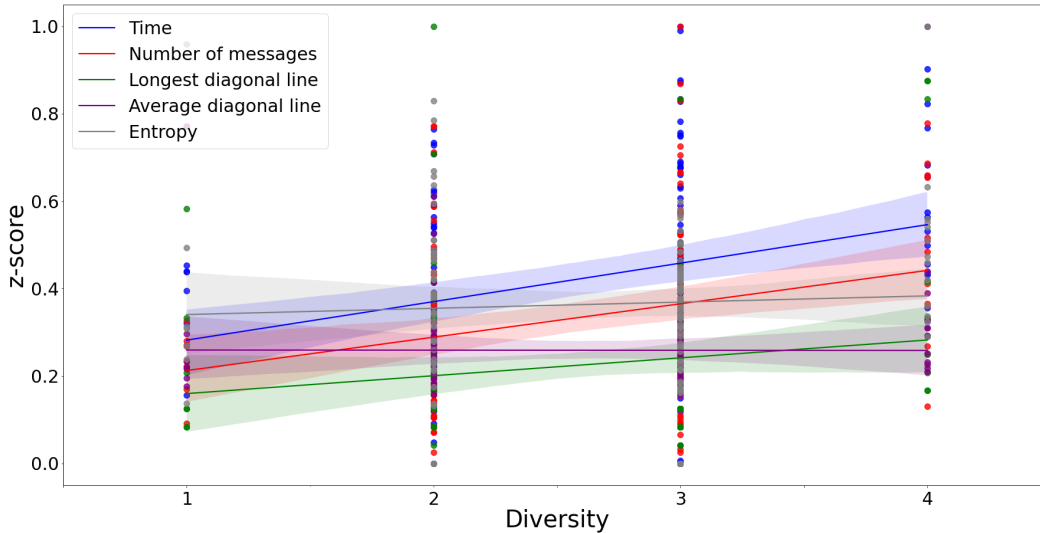


Figure 3: Normalized values of quantified measures of communication plotted by cultural diversity. Each dot represents individual values for the five measures, along with the fitted regression line. Shaded regions around the lines represents the confidence intervals for each regression.

time spent in chat, complexity and unpredictability of recurrent patterns (entropy), average length of recurrent pattern (average diagonal line in recurrence plot), and longest recurrent pattern (longest diagonal line in recurrence plot). Regarding RQ2, our post-hoc analysis further reveal that groups with the highest degree of cultural diversity had diagnostically different structural patterns of communication compared to more homogeneous teams. Specifically, groups with the highest level of heterogeneity spent more time and sent more messages in chat, had greater complexity in interpersonal exchange, and lower divergence in group discourse. Notably, there was no sign of significant difference among the moderately diverse groups and homogeneous groups. We discuss the implications of results in greater details as follow.

First, the finding that groups with greater cultural diversity spent more time and sent more messages during the discussion could indicate positive or negative group dynamics. On the positive side, higher number of messages signals more opportunities for information sharing and exchange of ideas, which may suggest that members in these teams are more actively engaged in the collaborative discourse and provided their diverse perspectives to the problem. On the other hand, it might also indicate teams getting stuck or struggle to reach consensus. Teams that struggle with internal conflicts also tend to spend more time on communication to resolve disagreements. This issue is more signified in the context of a CPS ranking task. Greater cultural differences in teams might result in different preferences and priorities over different job candidates, apartments, party venue, and professor qualities [14]. As such, increased cultural diversity in teams could bring challenges in team communication that are reflected through frequencies of messages and time spent on discussion. This finding is consistent with the notion that

cultural diversity in teams can be a “double-edged sword”. It has been widely suggested that cultural diversity is associated with both positive and negative team outcomes [66]. Previous studies demonstrate that compared to culturally homogeneous groups, culturally diverse teams are associated with higher levels of innovation and creativity but are also prone to more conflicts, less effective communication, and less cohesion [66]. It is worth noting that the interpretation of the results has to be situated in the context of tasks. For instance, in open-ended tasks like brainstorming, longer time spent and more messages are typically indications of more innovative perspectives and creativity. However, in decision making tasks, especially when groups were under time constraints to arrive at a decision, more time spent on the task and greater amount of discourse may suggest it takes more effort for heterogeneous groups to build shared understanding and reach a solution collectively.

Our RQA analysis further revealed more culturally diverse teams were associated with more complexity in the distribution of recurrent patterns (higher entropy), longer recurrent patterns of interactions (higher average length of diagonal lines, and higher longest diagonal line). The increased complexity in culturally diverse teams can signal less rigidity, more adaptability, and higher responsivity in team interactions. These attributes have been found to be key for innovation, creativity, and information sharing in team communication [61]. For instance, Fusaroli and Tylén found entropy and average diagonal line of recurrences in transcripts between the dyads to have a positive association with their performance [25]. In conjunction with our first finding, longer discussion time and higher number of messages may provide more opportunities for diverse groups to display recurrent patterns, which may explain why diverse

groups have higher longest recurrent patterns. Interestingly, we found that the most culturally diverse groups have noticeably higher longest recurrent patterns and lower divergence, suggesting higher convergence within group discussion over time. This is in contrast to previous research which has typically associated high cultural diversity with divergent communication behavior [66]. We consider two potential explanations for how higher convergence was reached in team interactions within the most diverse teams. First, the impact of diversity on performance can be mediated through goal-orientation and information elaboration [51]. We hypothesize that longer and more active discussions in more diverse groups allowed much more opportunities for information elaboration which may account for a longer recurrent pattern in diverse teams. Moreover, the goal-oriented nature of CPS decision making task may cultivate more convergent collaborative interaction patterns. Second, higher entropy in more diverse groups indicates the higher complexity in the recurrent patterns. Therefore, although the longest recurrent pattern shows convergence in discussions of more diverse teams, higher entropy suggests less rigidity in the patterns of behavior which allows for less habitual patterns and more flexibility in social and cognitive interaction sequences. These findings indicate a potential increased willingness within cultural diverse teams to engage in active information exchange and leverage different perspectives during collaborative discussions. Taken together, this finding suggests highly diverse groups' flexibility in communication structure and openness to new information. High convergence indicated their capacity in perspective-taking as well as the elaborated processing of new information. Moreover, higher longest diagonal line could possibly be a sign of lengthier discussion and argumentation to reach a conclusion which could be a result of less effective communication. This could be meaningful characteristics to look for as teams across educational and professional settings are increasingly diverse. Further research would be needed to examine whether such pattern is associated with productive performance and psychological outcomes for participating members.

Our work serves as a starting point for future studies to leverage RQA in establishing the possible positive and negative links between cultural diversity, communication dynamics, and other post-collaboration measures such as performance and team satisfaction. Moreover, we contribute a structural view of the recurrences of CPS skills through sociocognitive processes in discourse. By taking a look into how the temporal sociocognitive processes that occur during collaborative interactions are shaped by cultural diversity during CPS tasks, we can begin to take a step towards understanding factors in learning environments that make CPS communication more or less effective. In addition to leveraging CPS frameworks such as [2] to characterize group discourse, future research may also operationalize group dynamics by capturing linguistic features based on the content of messages through syntactic and semantic similarity, i.e., Conceptual Recurrence Plots [3]. This analysis would not only allow for more complexity and possibilities in the way recurrence appears in group communication, but also potentially reveal more nuanced linguistic patterns with respect to cultural diversity in teams. Finally, our current study mainly concerns with the degree of cultural diversity, how-

ever there is also a need to investigate the communication behavior of specific demographic subgroups (i.e. female, underrepresented minorities) in CPS. Further investigation on the differences within these demographic groups could examine how these factors independently and in combination play a role in shaping diverse groups. We call for more efforts towards promoting inclusivity and equity in teams. Future studies along this line should aim to provide further insights on identifying group patterns that promote effective problem solving and meaningful experiences among diverse teams.

7. CONCLUSION

We focused on the impact of cultural diversity on group communication in CPS. Our novel approach leveraged RQA to study the dynamics in group communication. Applying RQA to analyze CPS discourse provides a means to uncover how groups with different degrees of diversity exhibit different patterns of behavior during the collaborative process. Understanding these behaviors is essential given the dynamic and interdependent nature of CPS tasks. Specifically, insight on how group dynamics impact culturally diverse groups has important implications on monitoring and improving diverse group communications. In sum, our study emphasizes the need to further our understanding of the role of diversity in group communication behaviors as teams share information, negotiate, and navigate the problem-solving task. Exploring the intricacies of these group dynamics sets a first step towards future research on understanding how these behaviors relate to group outcomes such as performance, psychological experiences, and group satisfaction. Furthermore, this study aligns with the agenda of promoting diversity and inclusivity in AI systems. Our findings could be meaningful for the researchers in the greater EDM community for applying such methods to diagnose diverse team dynamics, as well as further inquiring positioning AI's critical role in structurally complex collaborative processes across diverse teams.

7.1 Acknowledgments

This research was supported by The Gates Foundation (#INV - 000752) and The Andrew W. Mellon Foundation (1806-05902). The authors would like to thank the Next Generation Undergraduate Success Measurement Project team members for help with data collection. Thanks to Educational Testing Services for the experiment information.

8. REFERENCES

- [1] J. J. Andrews and D. N. Rapp. Benefits, costs, and challenges of collaboration for learning and memory. *Translational Issues in Psychological Science*, 1(2):182, 2015.
- [2] J. Andrews-Todd and C. M. Forsyth. Exploring social and cognitive dimensions of collaborative problem solving in an open online simulation-based task. *Computers in human behavior*, 104:105759, 2020.
- [3] D. Angus, A. Smith, and J. Wiles. Conceptual recurrence plots: Revealing patterns in human discourse. *IEEE transactions on Visualization and Computer Graphics*, 18(6):988–997, 2011.
- [4] D. Angus, A. E. Smith, and J. Wiles. Human communication as coupled time series: Quantifying

- multi-participant recurrence. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(6):1795–1807, 2012.
- [5] R. Arum, J. S. Eccles, J. Heckhausen, G. A. Orona, L. von Keyserlingk, C. M. Wegemer, C. E. Wright, and K. Yamaguchi-Pedroza. A framework for measuring undergraduate learning and growth. *Change: The Magazine of Higher Learning*, 53(6):51–59, 2021.
- [6] J. B. Bear and A. W. Woolley. The role of gender in team collaboration and performance. *Interdisciplinary science reviews*, 36(2):146–153, 2011.
- [7] S. T. Bell, A. J. Villado, M. A. Lukasik, L. Belau, and A. L. Briggs. Getting specific about demographic diversity variable and team performance relationships: A meta-analysis. *Journal of management*, 37(3):709–743, 2011.
- [8] C. A. Bowers, J. A. Pharmed, and E. Salas. When member homogeneity is needed in work teams: A meta-analysis. *Small group research*, 31(3):305–327, 2000.
- [9] J. Burrus, T. Jackson, N. Xi, and J. Steinberg. Identifying the most important 21st century workforce competencies: An analysis of the occupational information network (o* net). *ETS Research Report Series*, 2013(2):i–55, 2013.
- [10] W. Cade, N. Dowell, A. Graesser, Y. Tausczik, and J. Pennebaker. Modeling student socioaffective responses to group interactions in a collaborative online chat environment. In *Educational Data Mining 2014*. Citeseer, 2014.
- [11] Z. Cai, B. Eagan, N. Dowell, J. Pennebaker, D. Shaffer, and A. Graesser. Epistemic network analysis and topic modeling for chat data from collaborative learning environment. In *Proceedings of the 10th international conference on educational data mining*, 2017.
- [12] E. Care, C. Scoular, and P. Griffin. Assessment of collaborative problem solving in education environments. *Applied Measurement in Education*, 29(4):250–264, 2016.
- [13] P. Chopade, D. Edwards, S. M. Khan, A. Andrade, and S. Pu. Cpsx: Using ai-machine learning for mapping human-human interaction and measurement of cps teamwork skills. In *2019 IEEE International Symposium on Technologies for Homeland Security (HST)*, pages 1–6. IEEE, 2019.
- [14] T. Cox. *Cultural diversity in organizations: Theory, research and practice*. Berrett-Koehler Publishers, 1994.
- [15] R. Dale, R. Fusaroli, N. D. Duran, and D. C. Richardson. The self-organization of human interaction. In *Psychology of learning and motivation*, volume 59, pages 43–95. Elsevier, 2013.
- [16] N. Dowell, Y. Lin, A. Godfrey, and C. Brooks. Promoting inclusivity through time-dynamic discourse analysis in digitally-mediated collaborative learning. In *International Conference on Artificial Intelligence in Education*, pages 207–219. Springer, 2019.
- [17] N. M. Dowell, W. L. Cade, Y. Tausczik, J. Pennebaker, and A. C. Graesser. What works: Creating adaptive and intelligent systems for collaborative learning support. In *International conference on intelligent tutoring systems*, pages 124–133. Springer, 2014.
- [18] N. M. Dowell, Y. Lin, A. Godfrey, and C. Brooks. Exploring the relationship between emergent sociocognitive roles, collaborative problem-solving skills, and outcomes: A group communication analysis. *Journal of Learning Analytics*, 7(1):38–57, 2020.
- [19] N. M. Dowell, T. M. Nixon, and A. C. Graesser. Group communication analysis: A computational linguistics approach for detecting sociocognitive roles in multiparty interactions. *Behavior research methods*, 51(3):1007–1041, 2019.
- [20] N. M. Dowell and O. Poquet. Scip: Combining group communication and interpersonal positioning to identify emergent roles in scaled digital environments. *Computers in Human Behavior*, 119:106709, 2021.
- [21] R. J. Ely and D. A. Thomas. Cultural diversity at work: The effects of diversity perspectives on work group processes and outcomes. *Administrative science quarterly*, 46(2):229–273, 2001.
- [22] S. M. Fiore, A. Graesser, and S. Greiff. Collaborative problem-solving education for the twenty-first-century workforce. *Nature human behaviour*, 2(6):367–369, 2018.
- [23] S. M. Fiore, K. A. Smith-Jentsch, E. Salas, N. Warner, and M. Letsky. Towards an understanding of macrocognition in teams: developing and defining complex collaborative processes and products. *Theoretical Issues in Ergonomics Science*, 11(4):250–271, 2010.
- [24] E. G. Foldy. Learning from diversity: A theoretical exploration. *Public Administration Review*, 64(5):529–538, 2004.
- [25] R. Fusaroli and K. Tylén. Investigating conversational dynamics: Interactive alignment, interpersonal synergy, and collective task performance. *Cognitive science*, 40(1):145–171, 2016.
- [26] J. C. Gorman, N. J. Cooke, P. G. Amazeen, and S. Fouse. Measuring patterns in team interaction sequences using a discrete recurrence approach. *Human Factors*, 54(4):503–517, 2012.
- [27] A. C. Graesser, N. Dowell, and D. Clewley. Assessing collaborative problem solving through conversational agents. In *Innovative assessment of collaboration*, pages 65–80. Springer, 2017.
- [28] A. C. Graesser, N. Dowell, A. J. Hampton, A. M. Lippert, H. Li, and D. W. Shaffer. Building intelligent conversational tutors and mentors for team collaborative problem solving: Guidance from the 2015 program for international student assessment. In *Building intelligent tutoring systems for teams*. Emerald Publishing Limited, 2018.
- [29] A. C. Graesser, S. M. Fiore, S. Greiff, J. Andrews-Todd, P. W. Foltz, and F. W. Hesse. Advancing the science of collaborative problem solving. *Psychological Science in the Public Interest*, 19(2):59–92, 2018.
- [30] J. Hao, L. Liu, A. A. von Davier, N. Lederer, D. Zapata-Rivera, P. Jakl, and M. Bakkenson. Epcal: Ets platform for collaborative assessment and learning. *ETS Research Report Series*, 2017(1):1–14, 2017.
- [31] D. A. Harrison, K. H. Price, and M. P. Bell. Beyond

- relational demography: Time and the effects of surface-and deep-level diversity on work group cohesion. *Academy of management journal*, 41(1):96–107, 1998.
- [32] A. C. Homan, J. R. Hollenbeck, S. E. Humphrey, D. V. Knippenberg, D. R. Ilgen, and G. A. Van Kleef. Facing differences with an open mind: Openness to experience, salience of intragroup differences, and performance of diverse work groups. *Academy of Management Journal*, 51(6):1204–1222, 2008.
- [33] S. K. Horwitz and I. B. Horwitz. The effects of team diversity on team outcomes: A meta-analytic review of team demography. *Journal of management*, 33(6):987–1015, 2007.
- [34] H. Jeong and M. T. Chi. Knowledge convergence and collaborative learning. *Instructional Science*, 35(4):287–315, 2007.
- [35] H. Jeong, C. E. Hmelo-Silver, and K. Jo. Ten years of computer-supported collaborative learning: A meta-analysis of cscl in stem education during 2005–2014. *Educational research review*, 28:100284, 2019.
- [36] A. P. Knight, D. M. Kennedy, and S. A. McComb. Using recurrence analysis to examine group dynamics. *Group Dynamics: Theory, Research, and Practice*, 20(3):223, 2016.
- [37] A. Kramer, D. P. Bhave, and T. D. Johnson. Personality and group performance: The importance of personality composition and work tasks. *Personality and Individual Differences*, 58:132–137, 2014.
- [38] P. C. Kyllonen, M. Zhu, and A. A. v. Davier. *Innovative assessment of collaboration*. Springer, 2017.
- [39] J. R. Landis and G. G. Koch. The measurement of observer agreement for categorical data. *biometrics*, pages 159–174, 1977.
- [40] S. S. Levine, E. P. Apfelbaum, M. Bernard, V. L. Bartelt, E. J. Zajac, and D. Stark. Ethnic diversity deflates price bubbles. *Proceedings of the National Academy of Sciences*, 111(52):18524–18529, 2014.
- [41] Y. Lin, N. Dowell, and A. Godfrey. Skills matter: Modeling the relationship between decision making processes and collaborative problem-solving skills during hidden profile tasks. In *LAK21: 11th International Learning Analytics and Knowledge Conference*, pages 428–437, 2021.
- [42] Y. Lin, N. Dowell, A. Godfrey, H. Choi, and C. Brooks. Modeling gender dynamics in intra and interpersonal interactions during online collaborative learning. In *Proceedings of the 9th international conference on learning analytics & knowledge*, pages 431–435, 2019.
- [43] L. Liu, J. Hao, A. A. von Davier, P. Kyllonen, and J.-D. Zapata-Rivera. A tough nut to crack: Measuring collaborative problem solving. In *Handbook of research on technology tools for real-world skill development*, pages 344–359. IGI Global, 2016.
- [44] Y. Lou, P. C. Abrami, and S. d’Apollonia. Small group and individual learning with technology: A meta-analysis. *Review of educational research*, 71(3):449–521, 2001.
- [45] L. Lu, Y. C. Yuan, and P. L. McLeod. Twenty-five years of hidden profiles in group decision making: A meta-analysis. *Personality and Social Psychology Review*, 16(1):54–75, 2012.
- [46] M. A. Marks, J. E. Mathieu, and S. J. Zaccaro. A temporally based framework and taxonomy of team processes. *Academy of management review*, 26(3):356–376, 2001.
- [47] N. Marwan, M. Carmen Romano, M. Thiel, and J. Kurths. Recurrence plots for the analysis of complex systems. *Physics Reports*, 438(5):237–329, 2007.
- [48] N. Marwan, N. Wessel, U. Meyerfeldt, A. Schirdewan, and J. Kurths. Recurrence-plot-based measures of complexity and their application to heart-rate-variability data. *Physical review E*, 66(2):026702, 2002.
- [49] D. McGunagle and L. Zizka. Employability skills for 21st-century stem students: the employers’ perspective. *Higher Education, Skills and Work-Based Learning*, 2020.
- [50] P. L. McLeod, S. A. Lobel, and T. H. Cox Jr. Ethnic diversity and creativity in small groups. *Small group research*, 27(2):248–264, 1996.
- [51] A. Nederveen Pieterse, D. Van Knippenberg, and D. Van Dierendonck. Cultural diversity and team performance: The role of team member goal orientation. *Academy of management journal*, 56(3):782–804, 2013.
- [52] OECD. Pisa 2015 collaborative problem solving framework, 2013.
- [53] P. Paulus. Groups, teams, and creativity: The creative potential of idea-generating groups. *Applied psychology*, 49(2):237–262, 2000.
- [54] V. Popov, O. Noroozi, J. B. Barrett, H. J. Biemans, S. D. Teasley, B. Slof, and M. Mulder. Perceptions and experiences of, and outcomes for, university students in culturally diversified dyads in a computer-supported collaborative learning environment. *Computers in Human Behavior*, 32:186–200, 2014.
- [55] O. Poquet, N. Dowell, C. Brooks, and S. Dawson. Are mooc forums changing? In *Proceedings of the 8th International Conference on Learning Analytics and Knowledge*, pages 340–349, 2018.
- [56] T. Rawald. *Scalable and Efficient Analysis of Large High-Dimensional Data Sets in the Context of Recurrence Analysis*. PhD thesis, Humboldt-Universität zu Berlin, Mathematisch-Naturwissenschaftliche Fakultät, 2018.
- [57] T. Reimer, A. Reimer, and U. Czienskowski. Decision-making groups attenuate the discussion bias in favor of shared information: A meta-analysis. *Communication Monographs*, 77(1):121–142, 2010.
- [58] D. C. Richardson, R. Dale, and N. Z. Kirkham. The art of conversation is coordination. *Psychological science*, 18(5):407–413, 2007.
- [59] M.-É. Roberge and R. Van Dick. Recognizing the benefits of diversity: When and how does diversity increase group performance? *Human resource management review*, 20(4):295–308, 2010.
- [60] M. Sarkar, T. Overton, C. Thompson, and G. Rayner. Graduate employability: Views of recent science graduates and employers. *International Journal of Innovation in Science and Mathematics Education*,

- 24(3), 2016.
- [61] M. C. Schippers, A. C. Edmondson, and M. A. West. Team reflexivity as an antidote to team information-processing failures. *Small Group Research*, 45(6):731–769, 2014.
- [62] B. Schneider, N. Dowell, and K. Thompson. Collaboration analytics—current state and potential futures. *Journal of Learning Analytics*, 8(1):1–12, 2021.
- [63] C. E. Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.
- [64] S. R. Sommers. On racial diversity and group decision making: identifying multiple effects of racial composition on jury deliberations. *Journal of personality and social psychology*, 90(4):597, 2006.
- [65] G. K. Stahl and M. L. Maznevski. Unraveling the effects of cultural diversity in teams: A retrospective of research on multicultural work groups and an agenda for future research. *Journal of International Business Studies*, 52(1):4–22, 2021.
- [66] G. K. Stahl, M. L. Maznevski, A. Voigt, and K. Jonsen. Unraveling the effects of cultural diversity in teams: A meta-analysis of research on multicultural work groups. *Journal of international business studies*, 41(4):690–709, 2010.
- [67] A. J. Strang, G. J. Funke, S. M. Russell, A. W. Dukes, and M. S. Middendorf. Physio-behavioral coupling in a cooperative team task: contributors and relations. *Journal of Experimental Psychology: Human Perception and Performance*, 40(1):145, 2014.
- [68] Y. Tausczik and X. Huang. The impact of group size on the discovery of hidden profiles in online discussion groups. *ACM Transactions on Social Computing*, 2(3):1–25, 2019.
- [69] H. Tenzer, M. Pudelko, and A.-W. Harzing. The impact of language barriers on trust formation in multinational teams. *Journal of International Business Studies*, 45(5):508–535, 2014.
- [70] D. C. Thomas. Cultural diversity and work group effectiveness: An experimental study. *Journal of cross-cultural psychology*, 30(2):242–263, 1999.
- [71] C. Tröster, A. Mehra, and D. van Knippenberg. Structuring for team success: The interactive effects of network structure and cultural diversity on team potency and performance. *Organizational Behavior and Human Decision Processes*, 124(2):245–255, 2014.
- [72] H. Van Dijk, M. L. Van Engen, and D. Van Knippenberg. Defying conventional wisdom: A meta-analytical examination of the differences between demographic and job-related diversity relationships with performance. *Organizational Behavior and Human Decision Processes*, 119(1):38–53, 2012.
- [73] M. Vigier and H. Spencer-Oatley. The interplay of rules, asymmetries in language fluency, and team dynamics in culturally diverse teams: Case study insights. *Cross Cultural & Strategic Management*, 2018.
- [74] A. A. von Davier. Computational psychometrics in support of collaborative educational assessments, 2017.
- [75] J. Wang, G. H.-L. Cheng, T. Chen, and K. Leung. Team creativity/innovation in culturally diverse teams: A meta-analysis. *Journal of Organizational Behavior*, 40(6):693–708, 2019.
- [76] W. E. Watson, K. Kumar, and L. K. Michaelsen. Cultural diversity’s impact on interaction process and performance: Comparing homogeneous and diverse task groups. *Academy of management journal*, 36(3):590–602, 1993.
- [77] S. S. Webber and L. M. Donahue. Impact of highly and less job-related diversity on work group cohesion and performance: A meta-analysis. *Journal of management*, 27(2):141–162, 2001.
- [78] C. L. Webber Jr and J. P. Zbilut. Dynamical assessment of physiological systems and states using recurrence plot strategies. *Journal of applied physiology*, 76(2):965–973, 1994.
- [79] C. L. Webber Jr and J. P. Zbilut. Recurrence quantification analysis of nonlinear dynamical systems. *Tutorials in contemporary nonlinear methods for the behavioral sciences*, 94(2005):26–94, 2005.
- [80] R. Whorton, A. Casillas, F. L. Oswald, A. Shaw, J. Burrus, K. Mattern, B. Naemi, and R. Roberts. Critical skills for the 21st century workforce. *Building better students: Preparation for the workforce*, pages 47–72, 2017.
- [81] A. W. Woolley, C. F. Chabris, A. Pentland, N. Hashmi, and T. W. Malone. Evidence for a collective intelligence factor in the performance of human groups. *science*, 330(6004):686–688, 2010.

Predicting Cognitive Engagement in Online Course Discussion Forums

Guher Gorgun
Educational Psychology
University of Alberta
Edmonton, AB, Canada
gorgun@ualberta.ca

Seyma N. Yildirim-Erbasli
Educational Psychology
University of Alberta
Edmonton, AB, Canada
seymanur@ualberta.ca

Carrie Demmans Epp
EdTeKLA, Computing Science
University of Alberta
Edmonton, AB, Canada
cdemmansepp@ualberta.ca

ABSTRACT

The need to identify student cognitive engagement in online-learning settings has increased with our use of online learning approaches because engagement plays an important role in ensuring student success in these environments. Engaged students are more likely to complete online courses successfully, but this setting makes it more difficult for instructors to identify engagement. In this study, we developed predictive models for automating the identification of cognitive engagement in online discussion posts. We adapted the Interactive, Constructive, Active, and Passive (ICAP) Engagement theory [15] by merging ICAP with Bloom's taxonomy. We then applied this adaptation of ICAP to label student posts ($N = 4,217$), thus capturing their level of cognitive engagement. To investigate the feasibility of automatically identifying cognitive engagement, the labelled data were used to train three machine learning classifiers (i.e., decision tree, random forest, and support vector machine). Model inputs included features extracted by applying Coh-Metrix to student posts and non-linguistic contextual features (e.g., number of replies). The support vector machine model outperformed the other classifiers. Our findings suggest it is feasible to automatically identify cognitive engagement in online learning environments. Subsequent analyses suggest that new language features (e.g., AWL use) should be included because they support the identification of cognitive engagement. Such detectors could be used to help identify students who are in need of support or help adapt teaching practices and learning materials.

Keywords

cognitive engagement, Coh-Metrix, discussion forums, natural language processing (NLP), online learning

1. INTRODUCTION

G. Gorgun, S. N. Yildirim-Erbasli, and C. D. Epp. Predicting cognitive engagement in online course discussion forums. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 276–289, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853149>

Educational theories and empirical studies have emphasized the importance of learner engagement [34]. Engaged learners are active; they invest time and effort during the learning process [44, 56, 66]. Empirical studies have corroborated the importance of engagement for student learning and well-being [39, 57, 74]. Studies have consistently found a significant association between student engagement and the cognitive and non-cognitive skills of students. For example, engaged students are more likely to be academically successful [89], retain knowledge [9], and show higher levels of critical-thinking [13] and problem-solving [61, 97]. Furthermore, engaged learners are more likely to show higher levels of school belongingness and socio-emotional well-being [22, 36, 73]. They are also less likely to drop-out [2, 3, 41] or demonstrate negative behaviors in school [62, 93]. Consistent with these general educational outcomes, a positive association between active participation in online discussion forums and academic achievement has been found [23, 51, 85]. Learner engagement, therefore, is a vital and predictive aspect of student learning and well-being [30].

Given that student engagement is a strong correlate of student achievement, researchers have long been interested in identifying student engagement levels to improve student success and learning processes. Identifying student engagement levels could be used to generate actionable feedback for students [52]. This feedback can be provided by instructors or through automated feedback that is delivered via learning dashboards or system features that nudge students to engage with the task.

However, identifying student (dis)engagement is difficult for instructors in online settings. In recent years, researchers have been interested in automating engagement identification. Automatically identifying students' engagement levels may also help instructors identify potentially at-risk students. Instructors may intervene to minimize the negative impact of disengagement on student learning. Furthermore, engagement detection could generate real-time feedback that instructors could use themselves. This critical feedback may be used to inform and evaluate the effectiveness of instructional practices employed in the class. Additionally, engagement detection could be embedded in instructor-facing analytics dashboards to help them understand student engagement in their classes [67]. Detecting that many learners are disengaged with the task could help instructors change their instructional strategy and practice, adjusting to meet the

needs of learners in their class [1, 68]. Providing instructors with information about learner engagement could also enable them to intervene in a timely manner [31].

Instructors can harness information that is easily obtained in face-to-face learning environments to observe student behavior and make inferences about how they are engaging. However, certain types of engagement (e.g., cognitive, affective) are not directly observable. Students may also feign engagement. Even if the form of engagement the instructor is interested in monitoring is easy to observe in person, that may not be the case in online settings. With the recent challenges that the COVID-19 pandemic has brought and the associated widespread use of online learning environments [69], the automated detection of learner engagement has become more vital yet more complicated. With the use of log data and dashboards, instructors could evaluate learner time within the system or time on task as an indication of behavioral engagement [67]. Unfortunately, proxies, such as time spent within a system, will not help an instructor extract the type of cognitive engagement that students demonstrate in online learning environments.

Many have tried to define student engagement, which is latent and inherently multi-faceted. Along with this, they have tried to model different types of learner engagement (e.g., academic, behavioral, affective, cognitive) in various learning contexts (e.g., online education, in-person classes). Typically, engagement is defined as students' motivation, willingness, effort, and involvement in school or learning-related tasks [19, 87]. It is operationalized as students' investment of resources such as effort, time, and energy into learning [53, 60, 61, 62].

Behavioral, emotional, and cognitive engagement are frequently studied types of engagement. Behavioral engagement is generally defined as time on task [18, 42, 50, 64, 94], and focuses on overt student behaviors [7]. Emotional engagement typically refers to learners' feelings, affect, and emotional reactions [77, 81, 83]. Cognitive engagement is conceptualized as employing learning strategies and investing effort and persistence into the task at hand [18, 42, 86, 94]. Different types of engagement are not mutually exclusive. For example, some researchers have argued that cognitively engaged learners are also behaviorally engaged, but the reverse is not necessarily true [50].

Cognitive engagement is one of the most challenging types of engagement to detect. Although there are certain theories developed for characterizing and capturing cognitive engagement in physical classroom settings (e.g., ICAP; [15]), they can be difficult to directly apply in online discussion environments, particularly in discussion forum posts (see [90]).

In this study, we focused on developing predictive models for automating cognitive engagement detection in discussion posts using natural language processing (NLP) and machine learning (ML) methods. This will help us evaluate the feasibility of automating cognitive engagement identification in online discussion posts. We will also analyze the classification errors and feature importance when predicting cognitive engagement through these relatively transparent models. Understanding feature importance and classifica-

tion errors may help us develop better models for detecting cognitive engagement.

2. ENGAGEMENT IDENTIFICATION APPROACHES

Dewan and colleagues [30] proposed an engagement detection taxonomy and grouped engagement detection methods under three main categories: manual (e.g., self-report measures), semi-automatic (e.g., engagement tracing), and automatic (e.g., log file analysis).

Manual engagement detection methods include self-report measures, checklists, and rating scales. Using self-report measures, students are asked to indicate their own engagement levels (e.g., [5, 12, 43]). Although it is easier to administer and use self-report measures [82], one major limitation of these instruments is their susceptibility to reporter biases. When learners rate their own engagement level, they may not accurately reflect their engagement during learning processes [35] because they lack the capacity to do so, or students may not reflect their true engagement status to avoid getting in trouble in a class (e.g., [45]). Therefore, self-report measures may not provide an objective evaluation of engagement.

The checklists and rating scales that instructors use are less prone to self-report biases (e.g., self-serving bias) during the coding process. Yet, checklists and rating scales also have certain limitations. Using these methods requires a great deal of time and effort [30]. For example, instructors, especially in large-scale classrooms or online environments, may not track each learner accurately. Learners may also pretend to be on task when in fact they are not engaged with the task. Hence, teacher evaluations made through such means may not accurately present the engagement levels of learners in class. Additionally, both self-report measures and checklists may not accurately capture fluctuations in learner engagement [90].

Semi-automatic engagement detection methods include engagement tracing. Using log or trace data, the timing and accuracy of learner responses have been employed to identify engaged and disengaged learners [8, 54]. For example, an unrealistically short response time is considered an indication of student disengagement during a quiz attempt.

Finally, automatic engagement detection methods include sensor data, log data, and computer vision-based approaches that analyze eye movement, body posture, or facial expressions [30]. These detection methods may automatically extract features from learners' body movement or facial expressions using sensor data as input [91]. Additionally, learner activities have been traced in learning management systems to automatically extract features related to engagement detection [16, 17]. These approaches may provide useful real-time information about student engagement, but they introduce privacy concerns such as being recorded.

2.1 Cognitive Engagement Frameworks

Two frequently employed cognitive engagement frameworks include Community of Inquiry (CoI; [46]) and Interactive, Cognitive, Active, and Passive engagement (ICAP; [15]).

The CoI framework identifies three elements (i.e., social presence, teaching presence, and cognitive presence) that support successful learning. The cognitive presence element of the CoI framework has been widely used by researchers (e.g., [58, 70]) for analyzing student learning and developing predictive models in online courses. In those studies, cognitive presence encompasses five phases: triggering event, exploration, integration, resolution, and other.

The ICAP framework [15] conceptualizes hierarchical cognitive engagement levels. Higher levels are related to higher cognitive engagement and learning growth. From top to bottom, the order of cognitive engagement levels is interactive, constructive, active, and passive.

Although both frameworks have been used to model engagement, they approach the categorization of engagement differently and have distinct aims [38]. The CoI framework specifically targets online learning environments and tries to model how students develop ideas in online discussions. The ICAP framework has been used to characterize learning in both in-person and online learning environments. It focuses on students' *active learning* behaviors [15]. Both frameworks appear to be promising theoretical approaches for cognitive engagement modeling and prediction. CoI focuses on different phases whereas ICAP focuses on different levels of cognitive engagement.

In a recent study, Farrow and colleagues [38] compared these engagement frameworks to decipher their commonalities and differences. They found similarities between the predictors used for engagement detection (e.g., message length was correlated with higher levels of engagement in both frameworks) whereas there were differences in the interpretation of classes (e.g., ICAP rewarded interactivity more than CoI).

2.2 Identifying Engagement in Online Environments

Some studies of online learning have used qualitative content analysis methods [90, 98] to detect engaged and disengaged learners. These studies start with developing a coding scheme reflecting different levels of engagement (e.g., active or passive engagement). This coding scheme is then used by the trained coders to manually label learner activities. This labelling process requires that considerable time and manual labor be invested, which is why it is not surprising that automating engagement identification via machine learning methods has emerged as a viable approach.

There are several potential benefits of automating cognitive engagement detection that go beyond the reduction in human fallibility and manual labor. These potential benefits include reducing coding time, rapidly identifying at-risk students in terms of engagement, and the possibility of integrating learner engagement information into dashboards and learning management systems. Moreover, automation can build on the prior work that manually labelled student engagement and helped us to understand it at a smaller scale.

Automated engagement detection methods in online environments can be divided into two groups [56]. The first group of studies focuses on automatically extracting features

pertaining to learners' physical cues such as body movement, heart rate, head posture, or where learners are looking (e.g., [55, 65]). In a recent study, Li and colleagues [65] extracted students' facial features and trained a supervised model to classify student cognitive engagement type. Although facial features were found to be powerful predictors of cognitive engagement, these types of studies may require webcams to extract learners' facial features, body posture, or head postures. Additionally, the learners could be aware of the fact that they are being recorded, which can lead them to experience discomfort [29]. Furthermore, the implementation of such systems may be costly (e.g., integrating webcams) compared with extracting learner trace data from learning management systems.

The second group of studies focuses on using a less sensor-heavy approach. They extract linguistic features (e.g., coherence, number of words) and trace data (e.g., time on task, number of clicks) from online learning environments to detect engagement (e.g., [6, 58, 59, 70]). For example, Kovanović and colleagues [58] employed n-grams and part-of-speech features to train a predictive model of cognitive presence using the CoI framework [46]. Their model achieved 58.38% accuracy ($K = .41$). In a similar study, Kovanović and colleagues [59] extracted features using Coh-Metrix, Linguistic Inquiry and Word Count (LIWC), and latent semantic analysis (LSA) similarity to represent average sentence similarity. Their best model achieved 72% accuracy ($K = .65$). Moving from the COI lens to that of ICAP, Atapattu and colleagues [6] used word embeddings from Doc2Vec to detect only the active and constructive levels of cognitive engagement in a MOOC, where they ignored posts that were of a social nature. They argued that learners with active engagement posts paraphrased, repeated, or mapped resources whereas learners with constructive posts proposed new ideas or introduced external material going beyond what was covered in class [6]. Therefore, discussion posts similar to course content remained in close proximity to the vector space generated by Doc2Vec. These posts were classified as active engagement whereas discussion posts that were far away from the vector space were classified as constructive engagement.

Some linguistic and course-based contextual features might limit the generalizability of predictive models (e.g., n-gram based models are sensitive to vocabulary choices) of cognitive engagement across different courses and contexts. For example, in a recent study, Neto and colleagues [70] analyzed model generalizability across educational contexts by employing Coh-Metrix features and the CoI framework. In their study, the baseline model was trained with a biology course dataset and achieved 76% accuracy ($K = .55$). When applied to a dataset from a technology course, the model had an accuracy of 67% ($K = .20$), which indicates some limitations to the generalizability of the model to new courses.

3. PRESENT STUDY

The purpose of this study is to evaluate the feasibility of automating cognitive engagement identification in fully online graduate courses through the lens of the ICAP framework. Given that engagement is a latent trait, researchers typically start with creating labeled data so that they can train supervised ML models. Most studies have employed CoI for

labeling cognitive presence (e.g., [37, 58, 70]).

ICAP and CoI have different aims and operationalizations of cognitive engagement (see section 2.1). In this study, we defined cognitive engagement as investing effort and cognitive resources. ICAP is more congruent with our cognitive engagement definition since we want to identify different levels of engagement in posts. Therefore, we employed the ICAP framework as our theoretical background that informed labeling decisions.

There is a lack of consensus concerning the coding scheme of ICAP for online environments even though the available coding schemes partly overlap with one another. For example, while Atapattu et al. [6] employed a binary coding scheme (i.e., active vs constructive), Yogev et al. [95] employed a coding scheme with six categories.

We first adapted the available coding schemes for online discussion posts by aligning the engagement levels with Bloom’s taxonomy [4]. This was done to emphasize the increased nature of cognitive complexity in both the taxonomy and cognitive engagement. Then, we developed three supervised ML models to analyze the feasibility of cognitive engagement prediction in online discussion environments. We also analyzed feature importance to evaluate whether we identified the same order of features across the classifiers employed in this study.

Our research questions were as follows:

RQ1: To what extent can a model trained with Coh-Metrix and contextual features be used to automatically classify discussion posts based on cognitive engagement level?

RQ2: Which features are more important for cognitive engagement prediction?

RQ3: What types of misclassifications occur?

4. METHODS

The data for this study consists of discussion forums from fully-online, graduate-level courses. The course forums differed in terms of facilitation method (i.e., peer-facilitated vs instructor-facilitated) and course length, i.e., regular-term courses (long) and summer-term courses (short).

4.1 Participants and Study Context

The data used in this study were collected from an online discussion platform that is used to deliver courses in a highly ranked college of education in Canada. The data collection protocol was reviewed and approved by the university’s research ethics board prior to completion. Participant consent was obtained for supplementary data collection. The dataset included 4,217 posts that had been produced by 111 students. In Table 1, we present the number of students, term length, facilitation method, and percentage of cognitive engagement posts in each course.

The courses span departments within the college and cover topics from language learning to educational psychology, educational technology, and educational policy.

Table 1: Number of students, term length, facilitation method, and percentage (%) of cognitive engagement levels by course.

ID	<i>n</i>	Length	Method	S	A	C	I
1	31	Long	Instructor	57	24	17	2
2	18	Short	Instructor	10	30	41	19
3	17	Short	Peer	44	22	12	22
4	22	Long	Instructor	24	31	23	24
5	23	Long	Peer	11	29	17	43

Note. Social (S), Active (A), Constructive (C), Interactive (I)

Demographic data were not collected and the previous deidentification of the data means that it cannot be obtained retroactively. It is worth noting that students could have their data excluded at the post level by marking it as private.

4.2 Data Coding Procedure

Our coding scheme is largely informed by the coding framework developed by Wang et al. [90] and Yogev et al. [95]. We altered and simplified the coding scheme based on the challenges we observed during the coding process.

We informed the coding scheme with Bloom’s revised taxonomy [4] and adapted it to reflect the higher-order cognitive complexity in the ICAP framework. Specifically, we map Bloom’s taxonomy level indicators onto cognitive engagement indicators. Bloom’s taxonomy was chosen because it is widely used by K-12 and higher education instructors to develop measurable and observable instructional objectives, tapping different levels of cognitive complexity. Cognitive complexity refers to the amount of cognitive demand required to complete a task. Higher levels of cognitive engagement necessitate higher effort investment and align with higher levels of cognitive complexity in Bloom’s taxonomy. Hence, we emphasized the presence of higher-order skills when identifying higher levels of cognitive engagement in the coding scheme.

Figure 1 shows the cognitive engagement coding scheme we used for labelling the discussion forum posts.¹ It highlights how the hierarchical nature of cognitive engagement aligns with Bloom’s taxonomy levels. For example, in the lower levels of the taxonomy, we have *remembering* and *understanding*. Active engagement corresponds to these levels in the taxonomy since posts with active engagement include elements of paraphrasing, mapping resources, and retrieving the same or similar concepts that are covered in class. Constructive engagement aligns with the level of *applying* and *analyzing* in Bloom’s taxonomy since these posts compare, contrast, illustrate, and argue in a cause and effect fashion. Finally, Interactive engagement relates to the levels of *evaluating* and *creating* in the taxonomy because these discussion posts make judgments and evaluations about the topics covered.

Our coding scheme included four categories: social, active, constructive, and interactive. The engagement categories are hierarchical, and social engagement is the lowest cate-

¹We cannot share the post content due to learner privacy.

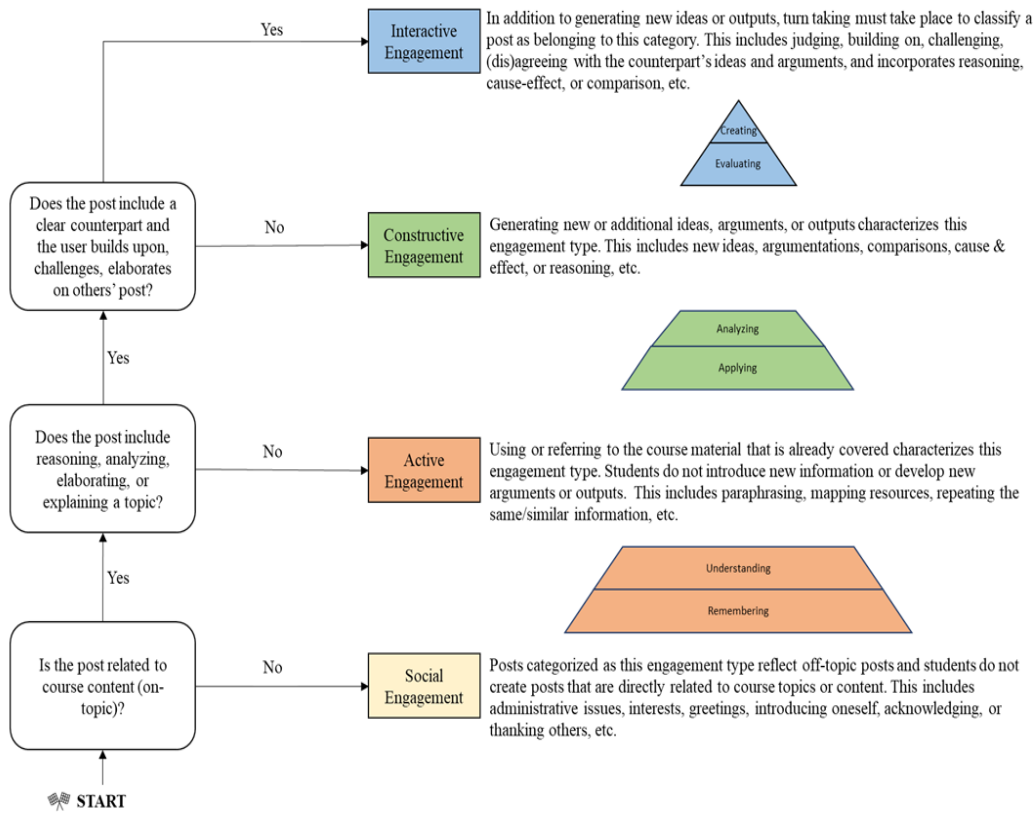


Figure 1: Cognitive engagement coding scheme.

gory followed by active, constructive, and interactive engagement. Note that we did not use *passive* engagement since our goal is not to categorize students’ engagement levels but rather to categorize the engagement observed in posts. Passive engagement entails that students read the posts but do not produce any posts, hence it cannot be employed for categorizing engagement in discussion posts. We also introduced a *social* engagement level corresponding to posts that are off-task and are meant to support relationship building (e.g., students introducing themselves).

Based on the coding scheme, we first identified whether the posts were on-task or off-task. Then, we asked whether the post exhibited reasoning, argumentation, or elaboration on a topic. Finally, we asked whether there was an evaluative argument and a clear counterpart in each post. While coding the posts, we assigned the highest engagement level observed in each post.

The coding was done by two content experts (Authors 1 and 2) who had previous experience with data coding and who were familiar with the cognitive engagement literature. First, we had a training session where we coded a sample of discussion posts, identified challenges, and reconciled discrepancies during the coding process. After the training session, we each coded a sample of discussion posts independently and we checked agreement between raters using the percent agreement method. We had an inter-rater reliability of .91.

Table 2: Distribution of cognitive engagement levels

ID	Label	Engagement Type	Number (%)
0		Social	1197 (28)
1		Active	1173 (28)
2		Constructive	930 (22)
3		Interactive	917 (22)

Table 2 shows the distribution of the four levels of cognitive engagement across all posts. The most frequently observed levels of cognitive engagement were social (28%) and active (28%). Given that constructive and interactive levels of engagement require higher cognitive effort and complexity, we expected to observe fewer constructive and interactive engagement level posts. Even though we expected to observe class imbalances based on previous studies (e.g., [38, 70]), we had a nearly balanced distribution of cognitive engagement levels. This might be due to the class structure where some instructors relied on peer facilitation as a management strategy for the course forums. Alternatively, having graduate-level courses could explain this balance.

4.3 Feature Extraction

This study used Coh-Metrix and non-linguistic contextual features to classify the cognitive engagement level seen in discussion posts.

Coh-Metrix is a tool used for discourse analysis. It estimates the cohesion, coherence, linguistic complexity, read-

Table 3: Hyperparameter search space used for model tuning

ML Classifier	Parameters	Values
Decision Tree	Max depth	3, 6, 9, 12, None
	Min samples leaf	A random integer between 1 and 10
	Max features	A random integer between 1 and 10
	Criterion	'Gini', 'Entropy'
Random Forest	Max depth	3, 6, 9, 12
	Min samples leaf	2, 4, 6, 10
	Max features	5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 107
	Number of estimators	500, 1000
	Min samples split	5, 10
	Criterion	'Gini', 'Entropy'
	Bootstrap	'True', 'False'
Support Vector Machine	Kernel	'linear', 'sigmoid', 'poly', 'rbf'
	C	50, 10, 1, 0.1

ability, and lexical category use in a text [32]. The English version of Coh-Metrix incorporates 108 features including referential cohesion, deep cohesion, and narrativity [33]. In previous studies on text classification, Coh-Metrix and contextual features were shown to be promising for cognitive presence identification (i.e., Community of Inquiry; [70]), suggesting their potential for supporting cognitive engagement detection in online discussion environments.

In addition to the Coh-Metrix features, we included three contextual features. These contextual features included information about whether a discussion post is a reply to another post, the number of replies that the post had received, and a count of the use of vocabulary from the academic word list (AWL) [20]. Whether a post is a reply was a binary variable, the number of replies a post received was an integer, and AWL count was an integer representing the total number of academic words in a post.

4.4 Data Pre-processing

To clean and prepare the corpus for training and classification, we ran several pre-processing steps. Our goal was to build a cognitive engagement classifier, thus we tried to create a corpus that was as clean and close to human readable form as possible [32]. We removed website links and “see attached” notations; stripped the html tags, eliminated new lines, white spaces, and tabs; expanded contractions; removed numbering and bullet points; and corrected misspelled words.

Because our unit of analysis was discussion posts, we created separate files for each post after applying the above data cleaning steps. We then ran Coh-Metrix 3.0 [47] to extract linguistic features for each post. All discussion posts have a single paragraph, so we removed the Coh-Metrix indicators of paragraph count (i.e., DESPC), standard deviation of the mean length of paragraphs (i.e., DESPLd), the mean of the LSA cosines between adjacent paragraphs (i.e., LSAPP1), and the standard deviation of LSA cosines between adjacent paragraphs (i.e., LSAPP1d). Finally, we created a data file containing all of the input features and the class labels. The input features included the 104 Coh-Metrix indicators and three other non-linguistic contextual features (e.g., whether a post is a reply). This data file was then used to train our

classifiers for cognitive engagement prediction.

4.5 Model Selection

To train and test models for predicting cognitive engagement, we split the dataset in two: 70% was used as a training set and the remaining 30% was used as a test set.

To answer our research questions, we trained three types of supervised classifiers:

- a **decision tree** (DT) [10] was selected because this approach is easier to interpret and the graphical representation of the tree can help us understand the relative importance of features;
- a **random forest classifier** (RF) [80] was chosen because it is an ensemble method that often exhibits superior performance on classification tasks in educational contexts (e.g., [59, 70]); and
- the **support vector machine** (SVM) [21] algorithm was chosen because it is designed to handle multidimensional data which may lead to superior performance when predicting cognitive engagement in discussion posts [40].

In addition to the above attributes, these types of models have previously performed well in other forum classification tasks [88] or they have supported educational data mining tasks with similarly sized or smaller data sets [25, 78]. Moreover, these classification algorithms are relatively transparent so they can aid us in understanding the contribution of each feature to cognitive engagement prediction.

4.6 Model Tuning

Model training, hyperparameter tuning, and analyses were conducted in Python (Version 3.8.8) using the sklearn [71] and mlxtend [75] packages.

We tuned the hyperparameters of each classifier using randomized search with 10-fold nested cross-validation. In Table 3, we summarize the hyperparameters and values used to tune the models for each classifier. For the decision tree,

Table 4: Model performance by model and cognitive engagement level (Cog. Engage.)

Cog. Engage.	Decision Tree				Random Forest				Support Vector Machine			
	P	R	F1	Acc. (%)	P	R	F1	Acc. (%)	P	R	F1	Acc. (%)
Social	.65	.70	.68	.71	.77	.74	.76	.73	.79	.83	.81	.83
Active	.50	.46	.48	.46	.56	.56	.56	.52	.60	.54	.57	.54
Constructive	.71	.58	.64	.62	.74	.69	.71	.68	.77	.73	.75	.73
Interactive	.55	.65	.59	.64	.60	.68	.64	.70	.64	.73	.68	.73
Full	.59	.58	.58	.60	.66	.66	.65	.66	.70	.71	.70	.71

the best model performance was obtained with max depth = 6, criterion = Gini, max features = 106, and min samples leaf = 7. For RF, the best model performance was obtained with max depth = 40, criterion = entropy, and number of estimators = 600. The best performing SVM model was obtained with kernel = linear and regularization (C) = 0.1.

The best model achieved 68% accuracy for the decision tree, 95% accuracy for random forest, and 85% for SVM on the training set.

4.7 Model Comparison and Analysis

We evaluated classifier performance with the test set using accuracy (Acc), Cohen’s Kappa (K), precision (P), recall (R), and F1 using 10-fold cross-validation. We use Landis and Koch’s guidelines to interpret Cohen’s Kappa, where values below .20 indicate slight agreement, values between .21 and .40 indicate fair agreement, values between .41 and .60 indicate moderate agreement, values between .61 and .80 indicate substantial agreement, and values greater than .81 indicate strong agreement [63].

We also statistically compared the model performance using Cochran’s *Q* test. Cochran’s *Q* test is a generalized version of McNemar’s test and it can be used to compare more than two classifiers [76]. The null hypothesis for Cochran’s *Q* test states that there is no difference between model classification accuracies. We also used McNemar’s test, a non-parametric statistical test, to perform the subsequent paired comparisons [76]. We report continuity corrected *p*-values for paired comparisons.

We analyzed feature importance across models to evaluate feature contribution towards cognitive engagement identification. For the decision tree, we evaluated feature importance with the Gini index. For random forest, we used the Mean Gini Decrease value to evaluate the features with the most explanatory power. For the support vector machine, after hyperparameter tuning with kernel and regularization parameter (C), we evaluated the feature importance by comparing the size of the support vector coefficients with one another.

4.8 Error Analysis Procedures

To better understand the classification errors, we analyzed the confusion matrix and the misclassified discussion posts. The confusion matrix allowed us to analyze the number of errors across models. We also compared the error rates (instead of the absolute number of errors) by dividing each value in the confusion matrix by the number of posts in the corresponding class and depicted the model performance in Figure 3. The rows represent the actual (human-assigned)

labels; the columns represent the predicted labels. This procedure allowed us to, first, identify which types of cognitive engagement tend to be misclassified as another type. Second, we compared the descriptive statistics of predictors of those misclassified posts to examine the reasons for misclassification.

5. RESULTS

5.1 RQ1: Model Performance

In Table 4, we provide the performance measures (i.e., precision, recall, F1 score, and accuracy) for all three models by cognitive engagement level. We also provide these measures for the full model (i.e., when all classes are being considered). Note that the accuracy of the zero-rule classifier for the full model would be 28.4%. As can be seen in Table 4, all of the models outperformed this simple baseline. Moreover, Cochran’s test revealed statistically significant differences between the classifiers we built, $Q = 55.68$, $p < .001$. We report the results of specific paired model comparisons below.

For the decision tree, we obtained 60% accuracy (K = .46) for the full model. If we consider the F1 scores for each level of cognitive engagement within the decision tree model, we can see that it did a relatively good job of predicting social and constructive engagement. Whereas, active engagement scored below .5 on all of precision, recall, and F1. While this is better than chance, these are the lowest performance measures observed across all three models.

The accuracy of the random forest classifier was higher than that of the decision tree (McNemar’s $\chi^2 = 21.70$, $p < .001$). However, this increased accuracy was not accompanied by a change in agreement-level (K was .55). Again, we consider the model’s relative performance across levels of cognitive engagement, which showed that it performed best when predicting social and constructive engagement. It also performed relatively well for interactive engagement.

The support vector machine classifier outperformed both the decision tree (McNemar’s $\chi^2 = 61.31$, $p < .001$) and the random forest (McNemar’s $\chi^2 = 19.04$, $p < .001$) models on the full prediction task. The SVM model’s Kappa value (.61) suggested substantial agreement between the predicted and human-assigned labels [63]. Similar to the models trained with the random forest and decision tree algorithms, we obtained the best model performance for social and constructive engagement followed by interactive and active engagement.

5.2 RQ2: Feature Importance

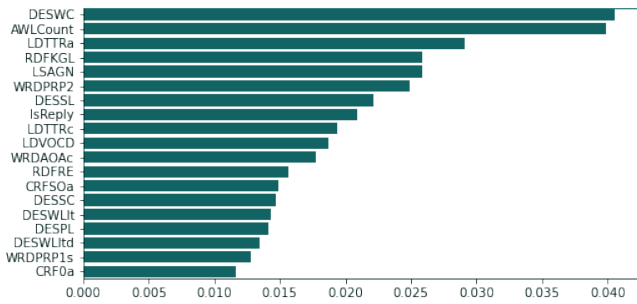


Figure 2: The feature importance scores for the 20 most important features for cognitive engagement prediction using random forest.

To evaluate feature importance for predicting cognitive engagement and answer our second research question, we first analyzed the decision tree classifier. The classification tree suggested that the most important predictor was the academic word list count (AWL Count), followed by whether a post is a reply, Flesch-Kincaid grade level (i.e., Coh-Metrix indicator of RDFKGL), and number of words (i.e., DESWC from Coh-Metrix).

To interpret the random forest classifier, we used the mean decrease in Gini coefficient because it supports the evaluation of the contribution of each feature to the model’s prediction [48]. Higher values of mean decrease in Gini score indicate that the feature contributed more when performing the prediction. Similar to the decision tree model, we found that the most important features were the number of words (i.e., DESWC), number of academic word list items (AWLCount), type-token ratio for all words (LDTTRa), and Flesch-Kincaid grade level (RDFKGL). Figure 2 shows this model’s most important twenty features and their relative weights.

For the support vector machine model, we evaluated the coefficient importance and found that the most important feature was second language readability (RDL2). This feature was followed by type-token ratio (LDTTRc), the standard deviation of the number of syllables in words (DESWLsyd), and LSA overlap between verbs (SMCAUSlsa).

We found that the decision tree and random forest models identified the same features as most important, yet their relative contribution to the prediction task changed across models. The support vector machine, on the other hand, identified a different set of features as most important.

5.3 RQ3: Error Analysis

All classifiers provided better prediction performance for social and constructive engagement (see Table 4). Table 5 presents the confusion matrix for the classifiers. The superior performance of the support vector machine model is evident in Table 5, where the misclassification of posts was the lowest across engagement categories.

We analyzed the performance measures and evaluated precision, recall, and F1 scores for each classifier and engagement category. This showed the jump between the precision and

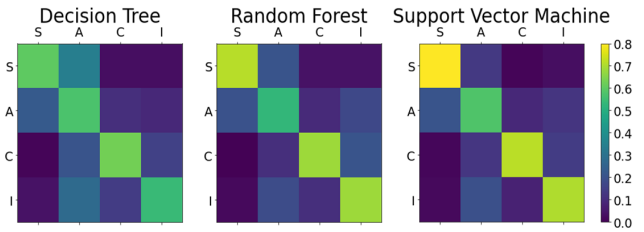


Figure 3: Classification error rates for decision tree, random forest, and support vector machine.

recall measures for constructive and interactive levels for the decision tree. For decision tree, the true positive rate was 65% for the interactive posts; that is, the classifier missed 35% of the interactive posts that should have been labelled with interactive. On the other hand, 45% of the posts were incorrectly identified as interactive. Furthermore, the true positive rate was 71% for the constructive posts. That is, the decision tree missed 29% of the constructive posts that should have been labelled with constructive, and 58% of the posts with other engagement levels were incorrectly identified as constructive.

The length of the posts and the smaller AWL count seem to be the primary reasons why social and active engagement posts were confused with each other. For misclassified social engagement posts, the mean post length ($M_S = 134$), AWL count ($M_S = 19.25$), and Flesch-Kincaid grade level ($M_S = 12.4$) were similar to the mean post length ($M_A = 138$), AWL count ($M_A = 12.93$), and Flesch-Kincaid grade level ($M_A = 11$) of active engagement posts. For the misclassified interactive posts, we observed a similar pattern. The mean post length ($M_I = 129.5$), AWL count ($M_I = 17.9$), and Flesch-Kincaid grade level ($M_I = 12.65$) were similar to those extracted from the active posts.

Additionally, the confusion matrix (Table 5) suggested that all classifiers tend to misclassify social engagement with active engagement. Of those misclassified posts, approximately 70% of social posts were classified as active engagement across the predictive models. That is, social engagement was incorrectly grouped into the higher level. The active engagement level tended to be misclassified as social engagement, indicating model confusion between adjacent engagement categories. Of the misclassified active posts, approximately 55% of them were classified as social engagement across all predictive models. In contrast, constructive engagement tended to be misclassified as interactive engagement across all classifiers. Of the misclassified constructive posts, approximately 40% of them were classified as interactive engagement across all predictive models. Effectively, this means that it was incorrectly grouped into the higher level. Misclassifications of interactive engagement went in the opposite direction and were not for the neighbouring class. Rather, these errors were classified as active engagement instead of their being recognized as interactive posts. Of the misclassified interactive posts, approximately 60% of them were classified as active engagement across all predictive models. One possible explanation for this misclassification is as follows: Active and interactive discussion posts tended to be shorter in length compared with those at the

Table 5: Confusion Matrix for Decision Tree, Random Forest, and Support Vector Machine

		Decision Tree				Random Forest				Support Vector Machine			
		S	A	C	I	S	A	C	I	S	A	C	I
Predicted	Social (S)	244	72	11	22	258	66	12	13	289	44	6	10
	Active (A)	89	158	33	62	69	190	30	53	67	185	36	54
	Constructive (C)	22	38	175	65	1	33	206	60	5	30	218	47
	Interactive (I)	18	49	29	179	6	53	29	187	4	47	23	201

constructive engagement level. Although constructive posts, in general, had higher word counts ($M_C = 339$) and AWL counts ($M_C = 40.3$), some of the misclassified active posts were shorter. This conflicted with the fact that constructive posts expanded course content with reflections and argumentation and were thus expected to be longer with greater use of academic vocabulary. This expectation did not hold in all cases and may explain why constructive engagement posts were misclassified as interactive.

To further evaluate misclassification errors for each classifier, we plotted model error rates. Figure 3 shows the kinds of errors that the decision tree, random forest, and support vector machine made. Rows represent actual classes and columns represent predicted classes. Across all models, social, constructive, and interactive engagement were less likely to be misclassified with each other. For the model trained with SVM, active and constructive as well as the active and interactive engagement levels were less likely to be misclassified with one another.

6. DISCUSSION

The purpose of this study was to develop models that identify cognitive engagement in online discussion forums. Using the post features that were extracted with Coh-Metrix, we trained three classifiers. Employing transparent classifiers allowed us to more easily evaluate feature importance for cognitive engagement prediction. We also conducted an error analysis to better understand the classification errors across models. Below, we discuss the implications of our findings in the context of each research question.

6.1 Feasibility of Automating the Prediction of Cognitive Engagement

Of the three types of predictive models we trained (i.e., decision tree, random forest, and support vector machine), the support vector machine performed sufficiently well for it to be used to identify engagement levels in discussion posts.

Previous studies generally focused on cognitive engagement prediction using the random forest algorithm and the CoI framework. We used a different cognitive engagement framework (i.e., ICAP), and our best model (SVM) demonstrated similar accuracy (70.3%) and Cohen’s kappa values ($K = .63$) to those that used random forest under the CoI model of cognitive presence (e.g., [38, 59]).

The development of these models is the first step towards supporting instructors who want to be able to identify student engagement so that they can improve their teaching practices and intervene when student engagement levels are

low. These types of analytics fall into the category of those desired by instructors as they capture aspects of student post quality that are not currently available in most dashboards [1]. The models could be used to inform instructors about the engagement status of students in online course discussions. Given model performance, it would be important to effectively communicate uncertainty in its labelling of student posts [24]. Approaches, such as those suggested by Brooks and Greer [11], could be used to mitigate the risk of instructors relying on misclassified data so that they appropriately trust the output of the model and act in accordance with its limitations. For example, the system could identify the students who are disengaging to warn instructors so that they adjust their teaching and instructional practices. Additionally, such systems could be used to nudge students to better engage with tasks or to recommend posts that may enhance student engagement levels [14].

6.2 Feature Importance for the Prediction of Cognitive Engagement

By investigating how different features contribute to model performance, we can better understand the underlying phenomena which, in turn, will support the development of better predictive models for detecting cognitive engagement. Our study also provided further empirical evidence of feature importance, corroborating the findings of previous studies.

Both the decision tree and random forest feature importance analyses identified similar features (e.g., AWL count, Flesch-Kincaid grade level, word count). The features identified for the models in the present study were consistent with previous studies (e.g., [38, 59]). Similar to these previous studies, we found that higher levels of cognitive engagement were associated with longer messages. Our work builds on this by highlighting the importance of vocabulary use through the AWL count feature, which was not included in the other studies. Rather, those studies captured vocabulary through other means (e.g., Linguistic Inquiry and Word Count [84]). Our findings indicate that the use of academic vocabulary, like those in the AWL, that are not specific to a discipline (words such as hypothesis, conclude) support the identification of cognitive engagement. Moreover, their more general nature means that they should support generalization across courses at similar academic levels.

6.3 Interpreting Classification Errors

Classification errors occurred between the active and social engagement labels. Across all of the algorithms we used in our study, the worst performance was observed for active engagement. In a recent study, Farrow and colleagues [38] also found similar results for their prediction of active

engagement using random forest and the CoI framework. However, they did not perform an error analysis.

Our analysis of active and social engagement classes suggested that median word count, median AWL count, and Flesch-Kincaid grade level were similar for these two engagement levels. Furthermore, the dispersion of predictor values for active engagement was wide, overlapping with the range of predictors of other engagement levels. This helps explain what may have contributed to the poor performance of models for active engagement identification. Constructive engagement, on the other hand, had more distinct dispersion than other engagement levels, which could explain why the models were more successful when predicting constructive engagement. These results suggest the importance of having distinct engagement categories for successfully differentiating the vector space, thus achieving higher model performance and accuracy across engagement categories.

6.4 Limitations and Future Directions

Like all models, ours capture certain characteristics of student engagement and the decisions we made influence both their accuracy and the extent to which they are expected to generalize to other settings. We discuss these issues below and their potential for creating new opportunities for the development and use of models in online-learning settings.

The overall performance of the models suggests there is room for improvement. Nonetheless, these models can provide a snapshot of the engagement level in each post, providing an early signal of student engagement in online learning environments. This signal could help instructors to derive insight from this student-engagement data.

We labelled the data with the highest engagement level observed in a post. However, different cognitive engagement levels (e.g., social and constructive) may co-exist in a single post. While this type of coding approach is commonly employed [90], it may have limited the performance of our classifiers. It also prevents a more nuanced understanding of the types of cognitive engagement demonstrated by students. To support more nuanced representations of student engagement, future research can consider the co-existence of different engagement levels or how engagement relates to different areas of a course using something akin to aspect-based sentiment analysis [96].

Another limitation of our study is that we focused on identifying cognitive engagement in online discussion posts. That is, posts are classified rather than students. Because engagement may fluctuate based on course content or a weekly basis, we employed post-level analyses, which is a common practice in such research (e.g., [38, 70]). However, this may influence performance metrics. Future research can focus on how to derive an appropriate measure and representation of a student's cognitive engagement based on the varied levels of engagement that are exhibited across their posts. Perhaps more importantly, students who did not post to the discussion forums or those who only read posts and did not create messages (i.e., online listeners [92]) have not been included because there were no posts from these students. Alternative mechanisms need to be found to characterize their engagement through a listening lens. Since simply opening

a post is not enough to infer cognitive engagement, proxy measures may need to be developed. Behavioural patterns such as the ratio between the re-reading of posts and posts made [1, 72] or scan-rate [49] may provide reasonable signals of online listener engagement.

7. CONCLUSION

In this study, we gauged the feasibility of automating cognitive engagement prediction through NLP and ML methods. We first manually coded over 4,000 posts using the coding scheme we adapted from ICAP and Bloom's taxonomy. Then, we extracted linguistic and contextual features and trained three machine learning classifiers to predict the level of cognitive engagement demonstrated in a forum post. We obtained promising results with a support vector machine.

Now that we have models that can identify the highest level of cognitive engagement seen in a single post, we can start to consider how other factors might interact with student cognitive engagement. For example, we know that the course structure, length, and facilitation method influence student participation [72], language use [27], and social support [26, 72]. Additionally, the system used to deliver a course can influence student engagement [28, 79]. Future research could improve cognitive engagement identification by integrating such information.

This work makes several contributions, with some at the theoretical level and others at the empirical level. First, we mapped cognitive engagement levels onto Bloom's taxonomy. Bloom's taxonomy places cognitive complexity in hierarchical order, as does our ICAP-based cognitive engagement coding scheme. Second, this study illustrated the utility of different classifiers for cognitive engagement prediction in graduate-level online courses. Our analysis of the importance of model features and errors is consistent with previous studies; it confirmed the importance of word count for predicting cognitive engagement and revealed the importance of AWL count and Flesch-Kincaid grade level for predicting cognitive engagement. Future studies can include these features and use a support vector machine to develop a predictive model for cognitive engagement in online learning environments. Building on this work will enable the development of better models that can then be used to inform teaching and learning when online discussion forums are part of course delivery.

8. ACKNOWLEDGEMENTS

This work is supported in part by funding from the Social Sciences and Humanities Research Council of Canada and the Natural Sciences and Engineering Research Council of Canada (NSERC), [RGPIN-2018-03834].

9. REFERENCES

- [1] G. Akcayir, L. Farias Wanderley, C. Demmans Epp, J. Hewitt, and A. Mahmoudi-Nejad. Learning analytics dashboard use in online courses: Why and how instructors interpret discussion data. In M. Sahin and D. Ifenthaler, editors, *Visualizations and Dashboards for Learning Analytics*, pages 371–397. Springer International Publishing, Cham, 2021. Series Title: Advances in Analytics for Learning and Teaching.

- [2] C. Alario-Hoyos, M. Pérez-Sanagustín, C. Delgado-Kloos, M. Munoz-Organero, et al. Delving into participants' profiles and use of social tools in moocs. *IEEE Transactions on Learning Technologies*, 7(3):260–266, 2014.
- [3] K. L. Alexander, D. R. Entwisle, and C. S. Horsey. From first grade forward: Early foundations of high school dropout. *Sociology of Education*, 70:87–107, 1997.
- [4] L. W. Anderson and D. R. Krathwohl. *A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives*. Longman, 2001.
- [5] J. J. Appleton, S. L. Christenson, D. Kim, and A. L. Reschly. Measuring cognitive and psychological engagement: Validation of the student engagement instrument. *Journal of School Psychology*, 44(5):427–445, 2006.
- [6] T. Atapattu, M. Thilakarathne, R. Vivian, and K. Falkner. Detecting cognitive engagement using word embeddings within an online teacher professional development community. *Computers & Education*, 140:1–14, 2019.
- [7] R. S. Baker, A. T. Corbett, I. Roll, and K. R. Koedinger. Developing a generalizable detector of when students game the system. *User Modeling and User-Adapted Interaction (UMUAI)*, 18(3):287–314, Aug. 2008.
- [8] J. E. Beck. Engagement tracing: Using response times to model student disengagement. In C.-K. Looi, G. McCalla, B. Bredeweg, and J. Breuker, editors, *Artificial intelligence in education: Supporting learning through intelligent and socially informed technology*, pages 88–95, Amsterdam, The Netherlands, 2005. IOS Press.
- [9] G. Bonet and B. R. Walters. High impact practices: Student engagement and retention. *College Student Journal*, 50(2):224–235, 2016.
- [10] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Wadsworth, 1984.
- [11] C. Brooks and J. Greer. Explaining predictive models to learning specialists using personas. In *Proceedings of the Fourth International Conference on Learning Analytics and Knowledge, LAK '14*, page 26–30, New York, NY, USA, 2014. Association for Computing Machinery.
- [12] W. Cade, N. Dowell, A. Graesser, Y. Tausczik, and J. Pennebaker. Modeling student socioaffective responses to group interactions in a collaborative online chat environment. In *Proceedings of the 7th International Conference on Educational Data Mining*, pages 399–400, 2014.
- [13] R. M. Carini, G. D. Kuh, and S. P. Klein. Student engagement and student learning: Testing the linkages. *Research in Higher Education*, 47(1):1–32, 2006.
- [14] Z. Chen and C. Demmans Epp. CSCLRec: Personalized recommendation of forum posts to support socio-collaborative learning. In A. N. Rafferty, J. Whitehill, V. Cavalli-Sforza, and C. Romero, editors, *Thirteenth International Conference on Educational Data Mining (EDM 2020)*, pages 364–373, Fully Virtual, 2020. International Educational Data Mining Society.
- [15] M. T. Chi and R. Wylie. The icap framework: Linking cognitive engagement to active learning outcomes. *Educational psychologist*, 49(4):219–243, 2014.
- [16] M. Cocea and S. Weibelzahl. Log file analysis for disengagement detection in e-learning environments. *User Modeling and User-Adapted Interaction*, 19(4):341–385, 2009.
- [17] M. Cocea and S. Weibelzahl. Disengagement detection in online learning: Validation studies and perspectives. *IEEE transactions on learning technologies*, 4(2):114–124, 2010.
- [18] K. S. Cooper. Eliciting engagement in the high school classroom: A mixed-methods examination of teaching practices. *American Educational Research Journal*, 51(2):363–402, 2014.
- [19] L. Corno and E. B. Mandinach. The role of cognitive engagement in classroom learning and motivation. *Educational Psychologist*, 18(2):88–108, 1983.
- [20] A. Coxhead. A new academic word list. *TESOL Quarterly*, 34(2):213–238, 2000.
- [21] K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2(Dec):265–292, 2001.
- [22] P. A. Creed, J. Muller, and W. Patton. Leaving high school: The influence and consequences for psychological well-being and career-related confidence. *Journal of Adolescence*, 26(3):295–311, 2003.
- [23] J. DeBoer, A. D. Ho, G. S. Stump, and L. Breslow. Changing “course” reconceptualizing educational variables for massive open online courses. *Educational Researcher*, 43(2):74–84, 2014.
- [24] C. Demmans Epp and S. Bull. Uncertainty representation in visualizations of learning analytics for learners: Current approaches and opportunities. *IEEE Transactions on Learning Technologies*, 8(3):242–260, 2015.
- [25] C. Demmans Epp and K. Phirangee. Exploring mobile tool integration: Design activities carefully or students may not learn. *Contemporary Educational Psychology*, 59(2019):1–17, Oct. 2019.
- [26] C. Demmans Epp, K. Phirangee, and J. Hewitt. Student actions and community in online courses: The roles played by course length and facilitation method. *Online Learning*, 21(4):53–77, Dec. 2017.
- [27] C. Demmans Epp, K. Phirangee, and J. Hewitt. Talk with me: Student behaviours and pronoun use as indicators of discourse health across facilitation methods. *Journal of Learning Analytics*, 4(3):47–75, Dec. 2017.
- [28] C. Demmans Epp, K. Phirangee, J. Hewitt, and C. A. Perfetti. Learning management system and course influences on student actions and learning experiences. *Educational Technology, Research and Development (ETRD)*, 68(6):3263–3297, Dec. 2020.
- [29] C. Develotte, N. Guichon, and C. Vincent. The use of the webcam for teaching a foreign language in a desktop videoconferencing environment. *ReCALL*, 22(3):293–312, 2010.

- [30] M. Dewan, M. Murshed, and F. Lin. Engagement detection in online learning: a review. *Smart Learning Environments*, 6(1):1–20, 2019.
- [31] N. Diana, M. Eagle, J. Stamper, S. Grover, M. Bienkowski, and S. Basu. An instructor dashboard for real-time analytics in interactive programming assignments. In *Proceedings of the seventh international learning analytics & knowledge conference*, pages 272–279, 2017.
- [32] N. M. Dowell, A. C. Graesser, and Z. Cai. Language and discourse analysis with coh-matrix: Applications from educational material to learning environments at scale. *Journal of Learning Analytics*, 3(3):72–95, 2016.
- [33] N. M. M. Dowell and A. C. Graesser. Modeling learners’ cognitive, affective, and social processes through language and discourse. *Journal of Learning Analytics*, 1(3):183–186, 2014.
- [34] J. J. Duderstadt, D. E. Atkins, D. E. Van Houweling, and D. Van Houweling. *Higher education in the digital age: Technology issues and strategies for American colleges and universities*. Greenwood Publishing Group, 2002.
- [35] S. D’Mello, B. Lehman, R. Pekrun, and A. Graesser. Confusion can be beneficial for learning. *Learning and Instruction*, 29:153–170, 2014.
- [36] J. S. Eccles, C. Midgley, A. Wigfield, C. M. Buchanan, D. Reuman, C. Flanagan, and D. Mac Iver. Development during adolescence: The impact of stage–environment fit on young adolescents’ experiences in schools and in families (1993). *American Psychologist*, 48(2):90–101, 1993.
- [37] E. Farrow, J. Moore, and D. Gašević. Analysing discussion forum data: a replication study avoiding data contamination. In *Proceedings of the 9th international conference on learning analytics & knowledge*, pages 170–179, 2019.
- [38] E. Farrow, J. Moore, and D. Gašević. Dialogue attributes that inform depth and quality of participation in course discussion forums. In *Proceedings of the Tenth International Conference on Learning Analytics & Knowledge*, pages 129–134, 2020.
- [39] R. Ferguson and D. Clow. Examining engagement: analysing learner subpopulations in massive open online courses (moocs). In *Proceedings of the fifth international conference on learning analytics and knowledge*, pages 51–58, 2015.
- [40] M. Fernández Delgado, E. Cernadas García, S. Barro Ameneiro, and D. G. Amorim. Do we need hundreds of classifiers to solve real world classification problems? *Journal of Machine Learning Research*, 15(1):3133–3181, 1993.
- [41] J. D. Finn and D. A. Rock. Academic success among students at risk for school failure. *Journal of Applied Psychology*, 82(2):221–234, 1997.
- [42] J. A. Fredricks, P. C. Blumenfeld, and A. H. Paris. School engagement: Potential of the concept, state of the evidence. *Review of Educational Research*, 74(1):59–109, 2004.
- [43] J. A. Fredricks and W. McColskey. The measurement of student engagement: A comparative analysis of various methods and student self-report instruments. In S. L. Christenson, A. L. Reschly, and C. Wylie, editors, *Handbook of research on student engagement*, pages 763–782. Springer, Boston, MA, USA, 2012.
- [44] S. Freeman, S. L. Eddy, M. McDonough, M. K. Smith, N. Okoroafor, H. Jordt, and M. P. Wenderoth. Active learning increases student performance in science, engineering, and mathematics. *Proceedings of the national academy of sciences*, 111(23):8410–8415, 2014.
- [45] K. A. Fuller, N. S. Karunaratne, S. Naidu, B. Exintaris, J. L. Short, M. D. Wolcott, S. Singleton, and P. J. White. Development of a self-report instrument for measuring in-class student engagement reveals that pretending to engage is a significant unrecognized problem. *PloS one*, 13(10):1–22, 2018.
- [46] D. R. Garrison, T. Anderson, and W. Archer. Critical inquiry in a text-based environment: Computer conferencing in higher education. *The Internet and Higher Education*, 2(2-3):87–105, 1999.
- [47] A. C. Graesser, D. S. McNamara, M. M. Louwerse, and Z. Cai. Coh-matrix: Analysis of text on cohesion and language. *Behavior Research Methods, Instruments, & Computers*, 36(2):193–202, 2004.
- [48] H. Han, X. Guo, and H. Yu. Variable selection using mean decrease accuracy and mean decrease gini based on random forest. In *2016 7th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, pages 219–224. IEEE, 2016.
- [49] J. Hewitt, C. Brett, and V. Peters. Scan Rate: A New Metric for the Analysis of Reading Behaviors in Asynchronous Computer Conferencing Environments. *American Journal of Distance Education*, 21(4):215–231, Nov. 2007.
- [50] M. Hu and H. Li. Student engagement in online learning: A review. In *2017 International Symposium on Educational Technology (ISET)*, pages 39–43. IEEE, 2017.
- [51] J.-L. Hung and K. Zhang. Revealing online learning behaviors and activity patterns and making predictions with data mining techniques in online teaching. *MERLOT Journal of Online Learning and Teaching*, 4(4):426–437, 2008.
- [52] H. Iraj, A. Fudge, H. Khan, M. Faulkner, A. Pardo, and V. Kovanović. Narrowing the feedback gap: Examining student engagement with personalized and actionable feedback messages. *Journal of Learning Analytics*, 8(3):101–116, 2021.
- [53] M. Jacobi et al. *College Student Outcomes Assessment: A Talent Development Perspective*. ASHE-ERIC Higher Education Report No. 7. ERIC, 1987.
- [54] J. Johns and B. P. Woolf. A dynamic mixture model to detect student motivation and proficiency. In *21st National Conference on Artificial Intelligence (AAAI)*, pages 163–168, Boston, MA, USA, 2006.
- [55] A. Kapoor and R. W. Picard. Multimodal affect recognition in learning environments. In *Proceedings of the 13th annual ACM international conference on Multimedia*, pages 677–682, 2005.
- [56] S. N. Karimah and S. Hasegawa. Automatic engagement recognition for distance learning systems: A literature study of engagement datasets and methods. In *International Conference on*

- Human-Computer Interaction*, pages 264–276. Springer, 2021.
- [57] R. F. Kizilcec, C. Piech, and E. Schneider. Deconstructing disengagement: analyzing learner subpopulations in massive open online courses. In *Proceedings of the third international conference on learning analytics and knowledge*, pages 170–179, 2013.
- [58] V. Kovanović, S. Joksimović, D. Gašević, and M. Hatala. Automated cognitive presence detection in online discussion transcripts. In *CEUR Workshop Proceedings*, volume 1137, pages 1–4, 2014.
- [59] V. Kovanović, S. Joksimović, Z. Waters, D. Gašević, K. Kitto, M. Hatala, and G. Siemens. Towards automated content analysis of discussion transcripts: A cognitive presence case. In *Proceedings of the sixth international conference on learning analytics & knowledge*, pages 15–24, 2016.
- [60] G. D. Kuh. *The National Survey of Student Engagement: Conceptual framework and overview of psychometric properties*. Indiana University Center for Postsecondary Research, 2001.
- [61] G. D. Kuh. The national survey of student engagement: Conceptual and empirical foundations. *New directions for institutional research*, 141:5–20, 2009.
- [62] G. D. Kuh, T. M. Cruce, R. Shoup, J. Kinzie, and R. M. Gonyea. Unmasking the effects of student engagement on first-year college grades and persistence. *The journal of higher education*, 79(5):540–563, 2008.
- [63] J. R. Landis and G. G. Koch. The measurement of observer agreement for categorical data. *Biometrics*, 33:159–174, 1977.
- [64] E. S. Lane and S. E. Harris. A new tool for measuring student behavioral engagement in large university classes. *Journal of College Science Teaching*, 44(6):83–91, 2015.
- [65] S. Li, S. P. Lajoie, J. Zheng, H. Wu, and H. Cheng. Automated detection of cognitive engagement to inform the art of staying engaged in problem-solving. *Computers & Education*, 163:1–13, 2021.
- [66] B. D. Lutz, A. J. Barlow, N. Hunsu, C. J. Groen, S. A. Brown, O. Adesope, and D. R. Simmons. Measuring engineering students’ in-class cognitive engagement: Survey development informed by contemporary educational theories. In *2018 ASEE Annual Conference & Exposition*, pages 1–15, 2018.
- [67] L. P. Macfadyen and S. Dawson. Mining lms data to develop an “early warning system” for educators: A proof of concept. *Computers & Education*, 54(2):588–599, 2010.
- [68] I. Molenaar and C. A. Knoop-van Campen. How teachers make dashboard information actionable. *IEEE Transactions on Learning Technologies*, 12(3):347–355, 2018.
- [69] B. A. Motz, J. D. Quick, J. A. Wernert, and T. A. Miles. A pandemic of busywork: Increased online coursework following the transition to remote instruction is associated with reduced academic achievement. *Online Learning*, 25(1):70–85, 2021.
- [70] V. Neto, V. Rolim, A. Pinheiro, R. D. Lins, D. Gašević, and R. F. Mello. Automatic content analysis of online discussions for cognitive presence: A study of the generalizability across educational contexts. *IEEE Transactions on Learning Technologies*, 14(3):299–312, 2021.
- [71] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [72] K. Phirangee, C. Demmans Epp, and J. Hewitt. Exploring the relationships between facilitation methods, students’ sense of community and their online behaviours. *Special Issue on Online Learning Analytics. Online Learning Journal*, 20(2):134–154, 2016.
- [73] J. Pietarinen, T. Soini, and K. Pyhältö. Students’ emotional and cognitive engagement as the determinants of well-being and achievement in school. *International Journal of Educational Research*, 67:40–51, 2014.
- [74] A. Ramesh, D. Goldwasser, B. Huang, H. Daume III, and L. Getoor. Uncovering hidden engagement patterns for predicting learner performance in moocs. In *Proceedings of the first ACM conference on Learning@ scale conference*, pages 157–158, 2014.
- [75] S. Raschka. Mlxtend: Providing machine learning and data science utilities and extensions to python’s scientific computing stack. *The Journal of Open Source Software*, 3(24), Apr. 2018.
- [76] S. Raschka. Model evaluation, model selection, and algorithm selection in machine learning. *arXiv preprint arXiv:1811.12808*, 2018.
- [77] K. A. Renninger and J. E. Bachrach. Studying triggers for interest and engagement using observational methods. *Educational Psychologist*, 50(1):58–69, 2015.
- [78] C. Romero and S. Ventura. Educational Data Mining: a Survey from 1995 to 2005. *Expert Systems with Applications*, 33(1):135–146, July 2007.
- [79] C. P. Rosé and O. Ferschke. Technology support for discussion based learning: From computer supported collaborative learning to the future of massive open online courses. *International Journal of Artificial Intelligence in Education*, 26(2):660–678, June 2016.
- [80] M. Schonlau and R. Y. Zou. The random forest algorithm for statistical learning. *The Stata Journal*, 20(1):3–29, 2020.
- [81] D. J. Shernoff. *Optimal learning environments to promote student engagement*. Springer, 2013.
- [82] D. J. Shernoff, M. Csikszentmihalyi, B. Schneider, and E. S. Shernoff. *Student engagement in high school classrooms from the perspective of flow theory*, pages 475–494. Springer, 2014.
- [83] D. J. Stipek. *Motivation to learn: Integrating theory and practice*. Pearson College Division, 2002.
- [84] Y. R. Tausczik and J. W. Pennebaker. The psychological meaning of words: LIWC and computerized text analysis methods. *Journal of language and social psychology*, 29(1):24–54, 2010.
- [85] S.-F. Tseng, Y.-W. Tsao, L.-C. Yu, C.-L. Chan, and K. R. Lai. Who will pass? analyzing learner behaviors

- in moocs. *Research and Practice in Technology Enhanced Learning*, 11(1):1–11, 2016.
- [86] C. O. Walker and B. A. Greene. The relations between student motivational beliefs and cognitive engagement in high school. *The Journal of Educational Research*, 102(6):463–472, 2009.
- [87] C. O. Walker, B. A. Greene, and R. A. Mansell. Identification with academics, intrinsic/extrinsic motivation, and self-efficacy as predictors of cognitive engagement. *Learning and Individual Differences*, 16(1):1–12, 2006.
- [88] N. Wanas, M. El-Saban, H. Ashour, and W. Ammar. Automatic scoring of online discussion posts. In *Proceedings of the 2nd ACM workshop on Information credibility on the web*, WICOW '08, pages 19–26, New York, NY, USA, 2008. ACM.
- [89] M.-T. Wang and R. Holcombe. Adolescents' perceptions of school environment, engagement, and academic achievement in middle school. *American Educational Research Journal*, 47(3):633–662, 2010.
- [90] X. Wang, M. Wen, and C. P. Rosé. Towards triggering higher-order thinking behaviors in moocs. In *Proceedings of the Sixth International Conference on Learning Analytics & Knowledge*, pages 398–407, 2016.
- [91] J. Whitehill, M. Bartlett, and J. Movellan. Automatic facial expression recognition for intelligent tutoring systems. In *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–6. IEEE, 2008.
- [92] A. F. Wise, S. N. Hausknecht, and Y. Zhao. Attending to others' posts in asynchronous discussions: Learners' online "listening" and its relationship to speaking. *International Journal of Computer-Supported Collaborative Learning*, 9(2):185–209, June 2014.
- [93] J.-Y. Wu, J. N. Hughes, and O.-M. Kwok. Teacher–student relationship quality type in elementary grades: Effects on trajectories for achievement and engagement. *Journal of School Psychology*, 48(5):357–387, 2010.
- [94] E. Yazzie-Mintz and K. McCormick. *Finding the humanity in the data: Understanding, measuring, and strengthening student engagement*, pages 743–761. Springer, 2012.
- [95] E. Yogev, K. Gal, D. Karger, M. T. Facciotti, and M. Igo. Classifying and visualizing students' cognitive engagement in course readings. In *Proceedings of the Fifth Annual ACM Conference on Learning at Scale*, pages 1–10, 2018.
- [96] W. Zhang, X. Li, Y. Deng, L. Bing, and W. Lam. A survey on aspect-based sentiment analysis: Tasks, methods, and challenges, 2022.
- [97] Y. Zhao, S. Lin, J. Liu, J. Zhang, and Q. Yu. Learning contextual factors, student engagement, and problem-solving skills: A chinese perspective. *Social Behavior and Personality: an international journal*, 49(2):1–18, 2021.
- [98] E. Zhu. Interaction and cognitive engagement: An analysis of four asynchronous online discussions. *Instructional Science*, 34(6):451–480, 2006.

Investigating Temporal Dynamics Underlying Successful Collaborative Problem Solving Behaviors with Multilevel Vector Autoregression

Guojing Zhou
University of Colorado Boulder
594 UCB, Boulder, CO 80309
guojing.zhou@colorado.edu

Chen Sun
Johns Hopkins University
Baltimore, MD, 21218
csun44@jhu.edu

Robert G. Moulder
University of Colorado Boulder
594 UCB, Boulder, CO 80309
robert.moulder@colorado.edu

Sidney K. D’Mello
University of Colorado Boulder
594 UCB, Boulder, CO 80309
sidney.dmello@colorado.edu

ABSTRACT

In collaborative problem solving (CPS), people’s actions are interactive, interdependent, and temporal. However, it is unclear how actions temporally relate to each other and what are the temporal similarities and differences between successful vs. unsuccessful CPS processes. As such, we apply a temporal analysis approach, Multilevel Vector Autoregression (mIVAR) to investigate CPS processes. Our data were collected from college students who collaborated in triads via a video-conferencing tool (Zoom) to collaboratively engage a physics learning game. Video recordings of their verbal interactions were transcribed, coded using a validated CPS framework, and organized into sequences of 10-second windows. Then, mIVAR was applied to the successful vs. unsuccessful CPS sequences to build temporal models for each. A comparison of the models together with a qualitative analysis of the transcripts revealed six temporal relationships common to both, six unique to successful level attempts, and another eight unique to unsuccessful level attempts only. Generally, for successful outcomes, people were likely to answer clarification questions with reasons and to ask for suggestions according to the current game situation, while for unsuccessful CPS level attempts, people were more likely to struggle with unclear instructions and to respond to inappropriate ideas. Overall, our results suggest that mIVAR is an effective approach for temporal analyses of CPS processes by identifying relationships that go beyond a coding and counting approach.

Keywords

Collaborative Problem Solving, Multilevel Vector Autoregression, Temporal Analysis, Interaction Patterns

G. Zhou, R. Moulder, C. Sun, and S. D’Mello. Investigating temporal dynamics underlying successful collaborative problem solving behaviors with multilevel vector autoregression. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 290–301, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853137>

1. INTRODUCTION

Collaborative problem solving (CPS) is a process where multiple people pool knowledge, skills, and efforts to solve complex problems [1, 26], and can be an effective approach compared to working individually [27, 28]. For example, in one study, pairs of people performed better than single individuals in discovering scientific laws [27]. In recent decades, CPS has appeared in more and more contexts such as schools [26, 36], online learning [34] and military tasks [41], and is recognized as an essential 21st century skill [1, 26].

But what exactly does CPS entail? Swiecki et al. argued that CPS is an interactive, interdependent, and temporal process [41]. Specifically, a typical CPS process involves two types of interactions: 1) people-task interactions where people interact with the environment or tools to solve problems [3] and 2) people-people interactions where team members interact with each other (e.g. exchanging information or coordinating behavior) to facilitate taskwork [23]. Interdependence refers to the fact that people rely on other people’s contributions to complete tasks, such as incorporating other people’s work/ideas into the solution and getting help from other people [25, 21]. Temporality refers to the fact that actions are produced as a series of inter-connected steps, and the strength of these connections relates to the temporal distance of the actions (the larger the distance, the weaker the connection) [8]. Thus, actions taken by the team at a certain time have an impact on the actions that the team will take in the near future, but this impact diminishes as time goes on. For example, when a question is asked, there is an immediate increase in the likelihood that the question will be answered, but this likelihood decreases as time progresses.

Since CPS is interactive, interdependent, and temporal, people’s behaviors ostensibly affect how tasks will be explored and whether tasks will be completed successfully or unsuccessfully. Emerging research has focused on discovering relationships between team behaviors and problem solving success [42, 1, 17, 6, 39]. However, as reviewed below, most existing work has investigated CPS behaviors independently, without effectively considering the interaction and interdependence between them [42, 1, 17]. This raises the following questions that motivate our work: how do teammates’ be-

haviors temporally relate to each other and what are the temporal similarities and differences between successful vs. unsuccessful CPS processes?

Further, with rapid development of remote collaboration tools and the high risk imposed by COVID-19 for holding face-to-face meetings, more and more collaborations are carried out remotely. Although remote collaboration has many advantages, it also narrows the communicative bandwidth [29]. For example, gesturing to physical objects is much more complex than in face-to-face communication [40] and certain patterns of social-visual attention (e.g., mutual gaze) are disrupted [43]. Given the necessity and challenges of remote collaboration, we focus on investigating the temporal dynamics of behaviors in remote CPS contexts.

2. RELATED WORK

Our literature review mainly focuses on two aspects of CPS processes related to our study: 1) successful vs. unsuccessful CPS and 2) temporal analysis of CPS behaviors.

2.1 Successful vs. Unsuccessful CPS Processes

Prior research has applied several methods to investigate CPS processes, such as qualitative analysis [2], coding-and-counting [1, 17, 6], and temporal analysis [6, 39]. For qualitative analysis, Barron studied face-to-face collaboration while solving math problems [2] using data from two three-person teams whose verbal interactions during collaboration were coded with three distinct functions: 1) articulation of solutions, 2) repetitions of proposals, and 3) responses to proposals. A qualitative analysis on the conversations revealed that successful CPS exhibited more effective mutual exchanges, better joint engagement, and shared goals.

Coding-and-counting analyses mainly count the occurrence of particular behaviors/actions and relate them to a desired outcome [42, 1, 17, 6, 39]. For example, Tausczik et al. analyzed data from an online mathematical problem solving community (MathOverflow) [42]. They first human-coded the discussion data with five distinct behavioral indicators and then built regression models to discover the association between indicator occurrences and solution quality [42]. Results showed that clarifying questions, critiquing answers, revising an answer and extending an answer significantly predicted solution quality. Similarly, Andrews-Todd et al. applied coding-and-counting to analyze college students' chat texts generated while collaboratively solving simulated electronics problems [1]. Based on the coded behaviors, they categorized students into four groups by a 2×2 median split - social (high vs. low) \times cognitive (high vs. low). Analysis results showed that the low social - low cognitive group performed worse than the other three groups. Additionally, having at least one high social - high cognitive student in the team could increase performance.

A few studies applied basic temporal analyses of CPS behaviors in addition to coding-and-counting methods. For example, Hao et al. analyzed CPS skills at the unigram and bigram levels [17]. They manually coded participants' collaboration chat texts generated while completing simulation-based tasks with four skills: sharing ideas (Share), negotiating ideas (Negotiation), regulating problem solving (Regulation), and maintaining communication (Maintain). Re-

sults indicated that at the unigram level (i.e., individual skills), effective CPS processes contained more negotiation than ineffective processes. At the bigram level (e.g., Share-Negotiation), effective processes had more share \rightarrow negotiate, negotiate \rightarrow share, regulate \rightarrow share, and negotiate \rightarrow negotiate patterns.

In another study, Chang et al. applied coding-and-counting and lag sequential analysis to examine students' interactions in simulated CPS scenarios [6]. Participants collaborated as triads (teams of three) via a chat box to solve physics problems. The chat texts were coded based on the Programme for International Student Assessment (PISA) CPS framework [26]. A counting based analysis indicated that successful groups had a high percentage (out of the total number of actions taken by the group) of two actions: 1) monitoring and reflecting on what they have done and 2) discussing the actions to take. Temporal lag sequential analysis showed that some students who engaged in trial-and-error failed to solve the problem in the end, while those demonstrating effective reasoning were more likely to solve the problem successfully. Similarly, Sun et al. applied coding-and-counting together with a pattern analysis approach to examine remote collaboration while students played a physics learning game [39]. Audio recordings of their verbal communications were transcribed and coded based on a validated CPS framework [38]. A coding-and-counting analysis showed that certain actions (e.g. discussing appropriate ideas, confirming understanding) were predictive of success in the game. Pattern analysis examining the co-occurrence of actions within short temporal windows revealed the importance of forming interactive communications among team members to establish common ground and support each other.

In sum, results from prior research suggested that CPS success is associated with both the occurrences of individual actions [2, 42, 1, 17] and connections between actions [17, 6, 39]. However these studies mainly utilized coding-and-counting to examine CPS success [42, 1, 17], and the temporal analyses were limited [17, 6, 39]. Next, we review studies that go beyond basic counts of individual behaviors by investigating temporal relationships of CPS behaviors.

2.2 Temporal Analysis of CPS

Prior work on temporal analyses of CPS has mainly focused on detecting the connections between individual actions from team interaction sequences [19, 7, 41, 9, 30, 22]. For example, Kapur applied lag sequential analysis to discover the most frequent communication patterns that emerged in CPS discussions [19]. Participants in the study coordinated in triads via online text chat to solve physics problems. Two experimental conditions (well-structured vs. ill-structured problems) were compared. The resultant chat texts were coded based on the Functional Category System framework [31]. Then, lag sequential analysis was separately applied to each condition to compute the transition frequency between indicators in consecutive utterance pairs. A comparison of the two resulting models showed that the ill-structured condition had several temporal between-action connections that occurred at least twice as frequently as those in the well-structured condition, whereas the well-structured condition had no such connections. This is a finding that coding-and-counting approaches did not reveal.

Csanadi et al. applied epistemic network analysis to investigate collaboration and thought processes of problem solving [7] and compared it with a coding-and-counting analysis. In that study, pre-service teachers were asked to reason about a pedagogical problem either in pairs or individually (following a think aloud protocol). Audio recordings of the discussions were segmented into propositional units, and coded with eight distinct indicators. Then epistemic network analysis was applied by sliding a window of two utterances on the coded sequences to discover co-occurrences of indicators. Based on the co-occurrence frequencies, a weighted, undirected graph was generated to represent the connections between indicators. A comparison of the networks for dyads and individuals showed that these two conditions had different indicator co-occurrence patterns. Again, the coding-and-counting analysis did not reveal such a finding.

Similarly, Swiecki et al. also applied epistemic network analysis to investigate the verbal interactions between team members in simulated military training scenarios [41]. In that study, 16 Navy air defense warfare teams performed the detect-to-engage sequence, which detects and identifies vessels or tracks, assesses whether tracks are threats, and decides what actions to take. Each team consisted of six roles and the analysis focused on the interactions between the tactical action officer and the other five members. Audio recordings of the verbal interactions were transcribed and coded via an automated coding scheme (nCodeR) [24]. Two conditions using a standard system (control) vs. a decision support system (experimental) were compared. In the epistemic network analysis, a sliding window of 5 utterances was applied to discover the temporal co-occurrence of indicators. Results showed that in the control condition, the tactical action officers frequently sought out information, while in the experimental condition, they mainly contributed information about the tactical situation. Again, the coding-and-counting analysis did not reveal such a finding.

Besides examining coded sequences, researchers have also investigated other types of sequences such as transcripts [9] and log traces [30, 22]. For transcripts, Dowell et al. applied group communication analysis to detect roles that emerged during group collaboration [9]. The analysis relied on computational linguistic techniques to map utterances into a latent semantic space (a n -dimension numerical space) upon which six measures were defined to describe the profile of each individuals' verbal contributions (e.g. social impact, overall responsivity, newness). Then, k -means clustering was applied to the profiles to detect emerging roles, resulting in six clusters with unique characteristics (e.g. "high social impact, responsiveness, and internal cohesion"). For log trace analysis, Perera et al. used sequential pattern mining to examine the process of competing a group course project for software development [30]. In that study, students collaborated in an online learning environment which logged events of group wiki editing, ticket management, and version changes. The event sequence of each group was then segmented into shorter sequences and sequential pattern mining was used to count the frequency of consecutive events (patterns) for each team. The analyses identified patterns (e.g., "1 version activity by 1 author" followed by "1 ticket activity by 1 author") that can distinguish the well-performing teams from the ill-performing ones.

2.3 Research Questions and Contributions

As reviewed above, researchers are beginning to uncover the specific behaviors - both individually as well as behavioral sequences - that predict CPS outcomes. Prior studies indicate that there are meaningful and detectable temporal connections between people's verbal actions (or problem solving actions) in CPS processes [19, 7, 41, 9]. Further, temporal analyses are sometimes more effective than coding-and-counting analyses for understanding collaborations [19, 7]. In the current study, we continue this line of research by utilizing multi-level vector autoregression (mlVAR) for a temporal analysis of CPS behaviors. mlVAR is a network analysis approach that examines temporal connections between actions in consecutive windows via linear mixed-effects models, thereby accounting for clustered/nested data. Specifically, for each variable in window t , mlVAR builds a linear model that uses all the variables in the $t - 1$ window to predict its value. mlVAR has been successfully applied in many domains, such as patient-physician interactions [16] and symptom-psychopathology interplays [10]. However, to our knowledge, it has yet to be utilized to investigate collaborative problem solving behaviors.

Towards this goal, we analyzed data from a study where students collaborated in triads via video conferencing to play a physics learning game. Next, we transcribed and segmented recordings of students' verbal communication into utterances, human-coded each utterance based on a validated CPS framework [39] and aggregated the occurrence of indicators with 10-second windows. We then applied mlVAR to the resultant multivariate time series to generate temporal graphs depicting relationships among indicators. Our primary research question is what are the temporal similarities and differences between successful vs. unsuccessful CPS interactions. We investigate this question by building separate mlVAR models for multivariate behavioral sequences associated with successful vs. unsuccessful CPS outcomes (i.e., solving a game level vs. failing to solve a level). Then, we compare the two resultant networks to discover the similarities and differences and conduct a qualitative analysis to further examine what interactions may facilitate CPS and what are the challenges people may face during CPS.

Our work extends existing CPS research in several ways. Whereas temporal approaches have been applied to analyze CPS data, some previous studies only focused on logged system actions rather than verbal communications [30, 22], which provides limited insights into how the collaboration unfolds (e.g. how students share information and negotiate). Of the studies that applied temporal analysis on coded sequences, some only considered the co-occurrence of indicators in a sliding window (e.g. epistemic network analysis), without specifying the direction of the relations [7, 41, 39]. Others only focused on the 1-to-1 temporal connections between variable pairs at time $t - 1$ and t (e.g. lag sequential analysis) [17, 6, 19], rather than n -to-1 connections as an action can be predicted by multiple prior actions [8, 9].

The approach we use in this work, mlVAR, has a number of desirable statistical properties over other approaches. First, unlike lag sequential analysis, which examines the 1-to-1 temporal connections between variables, mlVAR uses *a set of variables at time $t - 1$* to predict a variable at time t ,

resulting in n-to-1 temporal connections. This allows it to discover temporal connections fully in context. Second, as a statistical procedure, mlVAR allows for uncertainty quantification of findings through the use of p-values and standard errors. Finally, mlVAR analysis is inherently multi-level, thereby statistically accounting for the fact that CPS data is generally clustered within teams. Failure to take such clustering into account may yield untrustworthy results due to phenomena such as the Simpson’s Paradox [20]. The multilevel nature of mlVAR also allows for the estimation of both team-specific networks and general networks representing the dynamics of the average team. Another novelty is that we employed a quasi-experimental matching procedure to balance the number of successful vs. unsuccessful instances across multiple factors (e.g. school, experimental block, time duration) to factor out their possible impacts. Finally, rather than emphasizing short-term temporal relationships between actions (i.e., using the previous utterance to predict the next), we combined behaviors across short windows spanning an average of three utterances to better capture aspects of the unfolding dialogue (conversations involving three participants).

3. DATA SET

The data were collected as part of a larger study [11] involving collaborative problem solving. Only aspects germane to the present work are reported here.

3.1 Participants

A total of 303 Students (56% female, average age = 22 years) from two large public universities participated in the study. Self-reported race/ethnicity information indicated that 47% of participants were Caucasian, 28% Hispanic/Latino, 18% Asian, 2% Black or African American, 1% American Indian or Alaska Native, and 4% “other”. Students were assigned to 101 triads based on their scheduling availability. Thirty participants from 18 teams indicated they knew at least one person in their team prior to study. Participants were compensated with a \$50 Amazon gift card (96%) or course credit (4%) after completing the study.

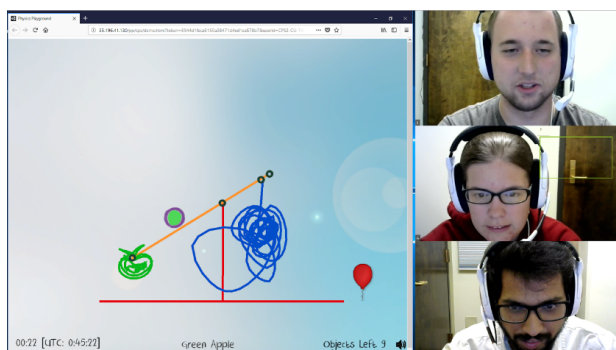


Figure 1: A screenshot of the collaboration scenario

3.2 CPS Task

Participants were tasked with collaboratively solving levels in a learning game “Physics Playground” [37], which is designed for young adults to learn Newtonian physics (e.g., Newton’s laws of force and motion). The goal for each level is to create objects and use physics laws to move a ball to a

designated target (a red balloon) as shown in Figure 1. Everything in the game obeys basic physics laws (e.g., there is gravity, each object has its own mass). As such, to move the ball in a desired way, students need to use simple machines such as levers and springboards. These, along with other objects (e.g., weights) are drawn using the mouse, upon which the new object becomes “alive” in the game and interacts with other existing objects following basic physics laws. Students can restart (clearing objects) or quit a game level at any time. They receive a gold coin if they solve a level with an optimal solution (i.e., with minimal objects), and they receive a silver coin for a sub-optimal solution (i.e., with more objects). No coin is rewarded for unsolved levels.

The game includes 17 levels covering two physics concepts: “energy can transfer” (EcT, 9 levels) and “properties of torque” (PoT, 8 levels). Each concept contains several subconcepts. For example, kinetic energy and gravitational potential energy belongs to EcT. The 17 levels varied in difficulty (as rated by two physics experts) and the levels were organized into three “playgrounds”, one per block in the study (detailed below). In the navigation page, students can choose a level to enter from the “playground” they are in or view the tutorials that introduce the game mechanics.

3.3 Procedure

The study involved an individual “at-home” part and a collaborative “in-lab” part. Materials (Qualtrics surveys) for the “at-home” part were emailed to students at least 24 hours prior to the lab session. It included several individual difference measures (e.g., prior knowledge, personality), a short tutorial on how to use Physics Playground, a short (around 15-mins) individual practice with the game, and other aspects unrelated to this study.

Students completed the “in-lab” part of the study in triads, using computer-enabled workstations equipped with a webcam and headset microphone. All collaborations occurred via the Zoom video-conferencing tool (as shown in Figure 1), and participants in the same group sat away from each other to avoid in-person interactions. The study involved three 15-min CPS blocks. (There was a fourth block for a different task not analyzed here.) In each block, one person was randomly assigned the role of a controller and the other two were tasked with being contributors. Assignment of the controller role rotated across blocks, so each teammate served as the controller for one of the three blocks. The game was loaded on the controller’s computer, so only the controller could directly interact with the game. The controller’s screen was shared with the contributors using the screen sharing feature of Zoom. Contributors participated in the problem solving process through verbal communications (e.g., proposing ideas, giving instructions).

The first block included five easy-to-medium levels involving a mix of EcT and PoT concepts. After that, all teams completed two 15-min experimental blocks, on either EcT or PoT levels (counterbalanced across teams) and with a specific CPS goal (delivered via verbal and on-screen messages): “solve as many levels as possible” or “get as many gold coins as possible”. The CPS goal and physics concept for the two experimental blocks were within-subjects factors, counter-balanced across teams.

Table 1: Facets and Indicators

Facet	Indicator
Constructing Shared Knowledge	1) Talks about challenge situations (Situations)
	2) Suggests appropriate ideas (ApporIdeas)
	3) Suggests inappropriate ideas (InapporIdeas) [N]
Negotiation and Coordination	4) Confirms understanding (Confirms)
	5) Interrupts others (Interrupts) [N]
	6) Provides reasons to support a solution (Reasons)
	7) Questions/Corrects others' mistakes (Questions)
	8) Responds to other's questions/ideas (Responds)
Maintaining Team Function	9) Criticizes, makes fun of, or being rude to other (Criticizes) [N]
	10) Discusses the results (DiscuResults)
	11) Brings up giving up the challenge (GivingUp) [N]
	12) Strategizes to accomplish task goals (Strategizes)
	13) Tries to quickly save an almost successful attempt (Saves)
	14) Asks others for suggestions (AskSuggest)
	15) Compliments or encourages others (Compliments)
	16) Initiates off-topic conversation (InitOffTopic)
	17) Joins in off-topic conversation (JoinOffTopic) [N]
	18) Provides instructional support (Instructions)
19) Apologizes for one's mistakes (Apologizes)	

3.4 Coding Collaboration Behaviors

We adopted the validated coding scheme developed by Sun et al. [39] to code student's verbal communications, which includes three main CPS facets: constructing shared knowledge, negotiation/coordination, and maintaining team function. Within each facet, there are several indicators that specify the concrete function(s) of each utterance as shown in Table 1 (see coding examples in Section 5.2). Constructing shared knowledge contains two aspects: a) disseminating knowledge, ideas, and resources among team members and b) establishing common ground for understanding the task and solutions [1, 26, 32]. Negotiation and coordination pertains to reaching a consensus on a solution plan to be carried out, such as dividing labor, resolving conflicts, integrating different perspectives, and monitoring execution [1, 18, 35]. Maintaining team function reflects efforts to maintain a functional team via assuming individual responsibilities, taking initiative, and co-regulation [5, 18, 33]. Most of the indicators in Table 1 describe positive (beneficial) CPS behaviors, and the negative behaviors were marked with N.

Coding was done on machine generated transcripts of students' verbal communications using IBM Watson's automatic speech recognition software [14]. IBM Watson both segments the audio into individual utterances and provides transcriptions of the utterances along with word timings. Utterances spoken by the same person within two seconds were merged to address segmentation errors (a 2-sec threshold was selected after considering a range of thresholds). The coders also viewed the video recordings of gameplay during coding to understand the context of students' interactions, nonverbal behaviors of the group dynamics, and to address speech recognition errors. An utterance was coded with an indicator if it contained evidence of the function described by the indicator. The coders marked the number of occurrences of each indicator per utterance, and multiple

indicators could occur for a given utterance (see Table 7 in Section 5.2 for an example). Three trained human coders coded the data. Each received two rounds of training before performing individual coding. After the second round of training, they all reached a high percentage of agreement (89% – 100%) and a high Gwet's AC1 value (0.91 to 1.00), a measure of interrater reliability specifically designed for cases of high agreement where more traditional metrics like Cohen's kappa yield unexpected results (sometimes called the paradoxes of kappa [15]).

3.5 Level Matching

Considering that besides collaboration behaviors, other factors such as level difficulty, the physics concept involved, and the problem solving goal (originally designed for other studies and not involved in our analysis) could also affect the problem solving outcome, we used a quasi-experimental matching procedure to factor out their possible impacts on level attempt success in our analysis. Specifically, an initial processing of the game logs yielded 1,164 level attempts (27% gold, 29% silver, and 44% no coin). Then, level attempts shorter than 60s were excluded, resulting in 808 level attempts. We removed short attempts because they are likely to be cases that students were exploring a level to decide whether to attempt it or not.

A preliminary analysis indicated that Energy can Transfer (EcT) levels (18% attempts succeeded, 7% earned a gold coin) were more difficult than Properties of Torque (PoT) levels (63% attempts succeeded, 40% earned a goal coin). To ensure a sufficient number of matches, matches for EcT levels focused on a coin (gold or silver) vs. a no coin comparison, whereas matches for PoT levels focused on gold vs. silver vs. no coin outcomes (i.e., a triplet). Matching was based on the following covariates: 1) school, 2) level identifier, 3) manipulation (i.e., gold coins vs. solve many levels), 4) block number (first or second) for the experimental blocks, and 5) duration of the level attempt. Level attempt duration was constrained to be at most 0.25 standard deviations of the mean duration of all the level attempts. An initial matching (using the "bmatch" function in the R package "designmatch" [46]) yielded 131 level attempt matches (33 Warmup, 69 EcT, and 29 PoT). Given the labor intensive nature of coding, 81 matches were randomly selected from the 131 candidates for analyses. The resulting data set included 209 level attempts: 66 (22 × 3) Warmup attempts from 47 unique teams, 68 (34 × 2) EcT attempts from 49 unique teams, and 75 (25 × 3) PoT attempts from 54 unique teams. A preliminary analysis on the resulting matches revealed that the matching indeed balanced school, manipulation, and block across the outcome groups (coins in this case). These data were coded for the CPS indicators as noted above. In our analysis, the level attempts that resulted in a coin (either gold or silver) were considered as successful attempts, whereas those resulted in no coin were treated as unsuccessful ones.

4. ANALYTICAL METHODS

4.1 Data Organization

Since students collaborated in triads in our studies, we aggregated the utterances into windows spanning an average of three utterances to investigate how their CPS interactions

Table 2: Average speaker changes and utterances with different window sizes

Size (Seconds)	Speaker changes	Utterances
8	1.604 (1.531)	2.559 (1.721)
9	1.850 (1.672)	2.849 (1.853)
10	2.087 (1.793)	3.150 (1.979)
11	2.340 (1.940)	3.457 (2.112)
12	2.587 (2.064)	3.741 (2.240)

unfolds. We used windows instead of utterance batches for the aggregation because the utterances varied in length and the windows can better balance the amount of information across units in the sequences. To find an appropriate window size, we tested 11 different sizes from 5 seconds to 15 seconds (sample shown in Table 2). As expected, the number of speaker changes (i.e., when the speaker changes between utterance $t+1$ and t) and number of utterances in each window increased for larger window sizes. We selected the window size of 10 seconds because it has an average of two speaker changes and three utterances in each window, which would accommodate one utterance for each of the three speakers, though windows ranging from 9 to 11 seconds would also have been suitable.

The aggregation sums up the occurrences of each indicator in each window, resulting in sequences of integer vectors. An utterance was assigned to a window if: 1) the center (midpoint) of the utterance lies in the window or 2) it overlaps with the window for more than one second (since the mean utterance duration was 2.08 seconds). Of the 19 indicators, 9 were exceedingly rare, occurring in less than 1 percent of the utterances. Accordingly, we merged these into a miscellaneous other indicator category (OtherIndi). We also grouped the utterances that were not coded with any indicators into a no indicator category (NoIndi). Finally, to reflect changes in the game, we applied a validated motion tracker tool [45] to capture changes in the game area. We used screen motion instead of logs to reflect game state because the logs only recorded the addition/deletion of objects, but not their interactions, and are generally limited for open-ended games where players can draw objects of any shape. The use of screen motion is also more generalizable and has been used in other studies investigating CPS [44]. In sum, our models contained 13 variables (12 indicator-related variables as shown in Table 3 + Screen Motion).

We conducted a few additional data processing steps prior to constructing the models. First, sequences shorter than 150s were excluded to ensure that each sequence has at least 13 transitions for our 13-variable models (detailed below). Second, windows shorter than 5 seconds (50% of normal window length) at the end of the sequences were removed because they were too short to be considered as a complete window, and then the last window of each sequence was removed. The second step removed the last 5 - 15 seconds of each level attempt to alleviate the concern that the language/indicators might have focused on the success or failure of the outcome rather than the problem solving process.

The final data set contains 133 level attempts (82 successful [silver or gold coin] and 51 unsuccessful [no coin]) from 74

Table 3: Average number of each indicator per window

Facet	Variable	Successful	Unsuccessful
Constructing	Confirms	0.30 (0.60)	0.33 (0.60)
Shared	ApporIdeas	0.23 (0.51)	0.14 (0.41)
Knowledge	InapporIdeas	0.15 (0.43)	0.21 (0.49)
	Situation	0.17 (0.49)	0.16 (0.49)
Negotiation and	Responds	0.33 (0.63)	0.31 (0.60)
	Reasons	0.14 (0.38)	0.11 (0.34)
Coordination	DiscuResults	0.09 (0.33)	0.08 (0.30)
Maintaining	Instructions	0.27 (0.60)	0.33 (0.65)
Team	Compliments	0.20 (0.48)	0.16 (0.42)
Function	AskSuggest	0.03 (0.17)	0.04 (0.21)
Other	OtherIndi	0.21 (0.50)	0.24 (0.54)
Variables	NoIndi	1.22 (1.31)	1.31 (1.30)

matches (detailed in section 3.5). For successful attempts, sequence length ranges from 14 to 85, with an average of 30.82 (SD = 15.93). Each window has 2.03 (SD = 1.79) speaker changes on average, 3.03 (1.89) utterances, and 17.08 (11.78) words. For unsuccessful attempts, sequence length ranges from 15 to 87, with an average of 36.92 (SD = 21.00). Each window has 2.12 (SD = 1.73) speaker changes on average, 3.14 (1.80) utterances, and 18.02 (11.42) words. In both data sets, each sequence has at least 13 transitions (minimal sequence length (14) - 1) for our 13-variable models. Table 3 shows the average number of each indicator per window, showing mean (SD).

4.2 Multilevel Vector Autoregression

Multilevel vector autoregression (mlVAR) is a network analysis method for understanding temporal dynamics between multiple variables nested within multiple higher order clusters (e.g., individuals or teams) [4, 13]. It is multilevel in that linear mixed-effects models are built to examine temporal connections at the individual level while accounting for group differences. Vector autoregression is the process of predicting a vector of variables at time t using the same vector at time $t - n$ (n is known as the lag). We utilized the R package “mlVAR” (version 0.5) for all analyses [12].

mlVAR analyzes the relations between temporal vectors by building a series of linear mixed-effects models, each of which uses the vector at time $t - n$ [$y(t - n)_1^i, y(t - n)_2^i, \dots, y(t - n)_J^i$] to predict an element of the vector at time t , denoted as $y(t)_j^i$, where i is subject id (in our case, level attempt id), j is variable id, and J is the length of the vectors (in our case, $J = 13$ as we have 13 variables). The mixed-effects models can be described by the following equations:

$$\begin{aligned}
 y(t)_1^i &= \mathbf{y}^i(\mathbf{t} - \mathbf{n})\mathbf{b}_1 + \mathbf{y}^i(\mathbf{t} - \mathbf{n})\mathbf{u}_1^i + e_1, \\
 y(t)_2^i &= \mathbf{y}^i(\mathbf{t} - \mathbf{n})\mathbf{b}_2 + \mathbf{y}^i(\mathbf{t} - \mathbf{n})\mathbf{u}_2^i + e_2, \\
 &\vdots \\
 y(t)_J^i &= \mathbf{y}^i(\mathbf{t} - \mathbf{n})\mathbf{b}_J + \mathbf{y}^i(\mathbf{t} - \mathbf{n})\mathbf{u}_J^i + e_J,
 \end{aligned} \tag{1}$$

where $\mathbf{y}^i(\mathbf{t} - \mathbf{n})$ is the $1 \times J$ vector at time $t - n$ for subject i , ($y(t - n)_1^i, y(t - n)_2^i, \dots, y(t - n)_J^i$); each of $\mathbf{b}_1, \dots, \mathbf{b}_J$ is a $J \times 1$ coefficient vector for the fixed-effects, which associates $\mathbf{y}^i(\mathbf{t} - \mathbf{n})$ to $y(t)_j^i$; each of $\mathbf{u}_1^i, \dots, \mathbf{u}_J^i$ is a $J \times 1$ random-

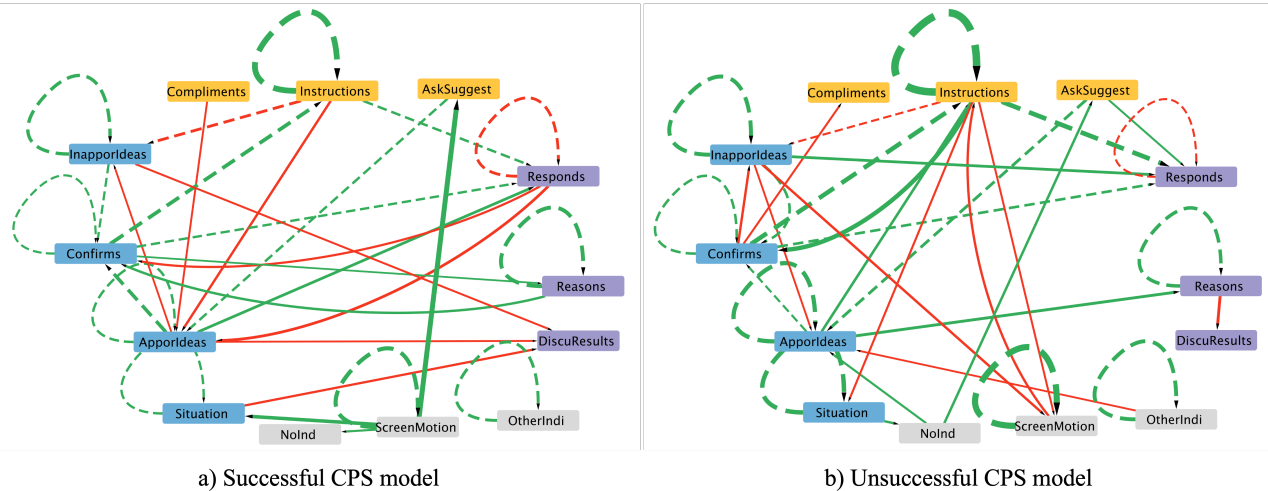


Figure 2: In the graphs, nodes were colored based on their facet, blue for constructing shared knowledge, orange for maintaining team function, purple for negotiation and coordination, and gray for other variables. Nodes with the same facet were grouped together to make within and between facet connections more obvious. Green lines in the graphs represent positive temporal relations and red lines represent negative relations. Dashed lines represent the relations that exist in both models, while solid lines represent the relations that exist only in one model. Arrows show the direction of the temporal relations and line thickness indicates the strength of the relationship (magnitude of the coefficient). Full names of the indicator labels are shown in Table 1.

effects vector, describing the deviation of individual i from the fixed-effects $\mathbf{b}_1, \dots, \mathbf{b}_J$; each of $\mathbf{e}_1, \dots, \mathbf{e}_J$ is a residual variable, describing the difference (error) between the model’s prediction and the actual value. Before estimating parameters for the linear mixed-effects models, all the variables are standardized using a z-transformation to ensure that the coefficient values are in the same range and comparable. We set the lag to be 1 as we were interested in fine-grained temporal effects (i.e., between 10-second consecutive windows). We also explored lag 2, but the resulting models had very few connections in them.

In mIVAR, a temporal connection exists between two variables if the “earlier” one $y(t-n)_j^i$ is a significant predictor of the “later” one $y(t)_j^i$ in the corresponding linear mixed-effects model. We used a p-value of .05 to denote a significant effect as other studies did [16, 10]. The temporal connections between variables can be represented by a directed, weighted graph (network), as shown in Figure 2.

5. RESULTS

Our temporal analysis of CPS processes consists of a quantitative analysis comparing the mIVAR models built on successful vs. unsuccessful level attempts and a qualitative analysis to interpret the patterns in the data.

5.1 mIVAR Analysis

For the successful and unsuccessful outcomes, the linear mixed-effects sub-models in the mIVAR model reached an average root mean squared error of 0.887 (SD = 0.062) and 0.906 (0.045) respectively in our standardized data. Figure 2 shows a comparison of the network for the successful vs. unsuccessful model. Since we are interested in whether there exist temporal connections between indicators rather than how strong the connections are, we treated all the sig-

nificant connections equally (indicated by the lines in the networks) and did not account for their strength (indicated by line thickness). As we can see, the graphs contain several kinds of relations, including self-loops, positive relations and negative relations. A positive self-loop suggests that the appearance of an action in window $t-1$ informs the appearance of the same kind of actions in window t . Generally, positive self-loops indicate that the indicator occurs across multiple consecutive windows. By contrast, a negative self-loop indicates that if an action appears in a window, it is less likely that the same kind of actions will appear in the next window. Given that the self-loops are easy to understand, we do not discuss them further.

Regarding between-indicator relations, most of the negative relations only exist in one of the two models. This is not surprising because actions can be absent for many reasons, such as being replaced by other actions or simply not being appropriate with respect to collaborative discourse (e.g., discussing the challenge situation after receiving a complement) or the stage in the problem solving process. As such, we mainly focus on the positive between-action relations.

Table 4 shows a summary of the positive between-action relations, where they were grouped into three categories based on whether they occurred in 1) both models (Common), 2) the successful model only (Successful), and 3) the unsuccessful models only (Unsuccessful). The “Common” category had 6 relations, 4 of which involved “confirms understanding”. Among them, “proposes appropriate ideas” and “proposes inappropriate ideas” were likely to influence “confirms understanding” questions, while “provides instructional support” and “responds to others’ ideas/questions” were likely reactions to “confirms understanding” questions. The other two relations “asks others for suggestions” → “proposes appropriate ideas” and “provides instructional support” → “re-

Table 4: Positive between-action connections

Common
Proposes appropriate ideas → Confirms understanding
Proposes inappropriate ideas → Confirms understanding
Confirms understanding → Provides instructional support
Confirms understanding → Responds to other’s questions/ideas
Asks others for suggestions → Proposes appropriate ideas
Provides instructional support → Responds to other’s questions/ideas
Successful
Confirms understanding → Provides reasons to support a solution
Provides reasons to support a solution → Confirms understanding
Screen motion → Asks others for suggestions
Screen motion → Talks about challenge situations
Screen motion → No indicators
Proposes appropriate ideas → Responds to other’s questions/ideas
Unsuccessful
Provides instructional support → Confirms understanding
Proposes appropriate ideas → Provides instructional support
Proposes appropriate ideas → Provides reasons to support a solution
Proposes inappropriate ideas → Responds to other’s questions/ideas
No indicators → Proposes appropriate ideas
No indicators → Asks others for suggestions
Talks about challenge situations → No indicators
Asks others for suggestions → Responds to other’s questions/ideas

sponds to others’ ideas/ questions” are ordinary interactions. The common relations suggest that the process of establishing shared understanding via asking and reacting to “confirms understanding” (clarification) questions was common to both successful and unsuccessful level attempts and likely underlies basic collaborative discourse.

Successful model: For the successful level attempts, there was a bidirectional temporal relation between “confirms understanding” questions and “provides reasons to support a solution”. Both reflect efforts to enhance shared understanding and thus the interactions between them may facilitate successful CPS. Importantly, we found that the degree of screen activity (“screen motion”) was a significant temporal predictor for three variables: “asks others for suggestions”, “talks about challenge situations”, and “no indicator utterances”. This suggests that people in successful CPS processes reflected on the current game situation. Among the three relations, “screen motion” → “talks about challenge situations” reflects that successful teams were linking what occurred in the game with the underlying challenge situation, which can be considered a form of metacognitive processing. The “screen motion” → “no indicator utterances” indicates that people were likely to talk following screen change but the language was not captured by the coding scheme since it might not have been CPS-related discourse. The “screen motion” → “asks others for suggestions” suggests that people were likely to ask for suggestions according to the current game state. Ostensibly, this relation connects people to the game and thus may facilitate CPS. The “proposes appropriate ideas” → “responds to others’ ideas/questions” reflects that people were likely to express their thoughts towards appropriate ideas, suggesting a form of affirming dialogue.

Unsuccessful model: In the “unsuccessful” model, “confirm understanding” questions were likely to appear following “provides instructional support”. This relation together with the “confirms understanding” → “provides instructional support” in the “common” category forms a bidirectional relation, which indicates that people were looping between giving instructions and confirming understanding. This loop suggests that people had difficulty with conveying and understanding instructions accurately.

There were three relations involving proposing ideas, both appropriate and inappropriate. The “proposes inappropriate ideas” → “responds to others’ ideas/questions” suggests that people were likely to discuss inappropriate ideas, which is different from the “proposes appropriate ideas” → “responds to others’ ideas/questions” in the “successful” model. This contrast suggests that devoting efforts to appropriate ideas rather than inappropriate ideas may help solve the problem. Additionally, “proposes appropriate ideas” → “provides instructional support” and “provides reasons to support a solution”. Given that providing instructions and reasons both give details to support an idea, these two relations suggest that the proposed ideas, though appropriate, needed further illustration or justification. The “no indicator utterances” (NoInd) were involved in three relations. Since these relations involved communication functions that were not described by the CPS coding scheme, we do not discuss them further. Finally, “asks others for suggestions” was a significant predictor of “responds to others’ ideas/questions”. We noticed that there was a common relation “asks others for suggestions” → “proposes appropriate ideas” in both the “successful” and “unsuccessful” models. This suggests that the most common reaction to suggestion requests is directly proposing an idea. Thus, reacting to suggestion requests by simply responding to others, rather than proposing new and task appropriate ideas, reflects that people were unsure about what to say, or how to proceed with solving the problem.



a) Confirms → Reasons b) Confirms ↔ Instructions

Figure 3: Screenshot for qualitative analysis examples

5.2 Qualitative Analysis

To further interpret the temporal patterns revealed by ml-VAR analysis, we conducted a qualitative analysis to examine what actually happened behind the patterns. Given the page limit, we focused on two important CPS factors: 1) establishing shared understanding, and 2) linking people to the game. As discussed above, shared understanding was often established via asking and reacting to “confirms understanding” (clarification) questions. Thus, we exam-

Table 5: “Screen motion” → “Asks others for suggestions” examples from successful level attempts

Example 1: Asking for suggestions after starting a level		
Time	Window	Speech/Actions
40-50	1	[controller actions]: Entered the game and made a few objects, but did not figure out how to solve the level. [Screen motion] value: 0.117 (> 99.32% instances)
54-58	2	<u>Controller</u> : Do you guys have any suggestions? [AskSuggest]
Example 2: Asking for suggestions after an attempt		
490-500	46	[controller actions]: Created a heavy weight and deleted it to spring the ball up, but it did not go high enough. [Screen motion] value: 0.010 (> 78.62% instances)
502-504	47	<u>Controller</u> : Should I try it again with, like Bigger? [AskSuggest]

Table 6: “Confirms understanding” ↔ “Provides reasons to support a solution” examples from successful level attempts

Example 1: “Confirms understanding” → “Provides reasons to support a solution”		
Time	Window	Speech/Actions
[Scenario]: The ball is dropping repeatedly and the team is trying to figure out how to capture and move it, as shown in Figure 3.a. (The key to solve this level is to catch the right timing.)		
317-319	17	<u>Contributor1</u> : When you see the ball start to drop, let go. [Instructions]
319-323	17	<u>Controller</u> : To drop and let go? [Confirms]
323-325	18	<u>Contributor1</u> : So when the ball begins falling on the screen-
325-325	18	<u>Controller</u> : Uh-huh.
325-328	18	<u>Contributor1</u> : Then stop drawing, [Instructions] ‘cause that’s when it’ll start to fall. [Reasons]
328-329	18	<u>Controller</u> : Okay. (silence) [Responds] [in-game action: the controller then solved the level]
Example 2: “Provides reasons to support a solution” → “Confirms understanding”		
[Scenario]: The team is trying to rotate an object.		
392-410	18-19	<u>Contributor1</u> : Oh. I would maybe try to put... now, um, get rid of those dots and um, put a new dot where um, the tip is. [ApporIdeas] So it could like move more forward. [Reasons]
410-411	19	<u>Controller</u> : Like here? [Confirms] [in-game action: pointed to a position using mouse]
411-417	19	<u>Contributor1</u> : Um, yeah. Like how you connected them. Connect them again at like, the bottom of the thing. [Instructions]

Table 7: “Confirms understanding” ↔ “Provides instructional support” example from an unsuccessful level attempt

Time	Window	Speech/Actions
[Scenario]: One contributor is trying to convey the idea of using a springboard to bounce the ball up (see Figure 3.b).		
281-291	26-27	<u>Contributor1</u> : Like, put it like under. Actually no, I feel like that’s too...No like, you have to like connect and like, and like, put it underneath. [Instructions] Like you feel-
291-292	27	<u>Contributor2</u> : What? (laughs). [Confirms]
292-307	27-28	<u>Contributor1</u> : Like, connect the two dots, then like put it under the ball. [Instructions] Because like, you’re going to delete that black line [Instructions] and then it’s going to fall on the...it’s going to end up falling on the bouncy board and hopefully go up. [Reasons]
307-308	28	<u>Controller</u> : Oh, wait. So go like this? [Confirms] [in-game action: drew a short line that connected two dots, but still did not make a springboard]
308-312	29	<u>Contributor1</u> : Yeah, and then make it go under, don’t hit that...um black. [Instructions]
312-313	29	<u>Controller</u> : Huh? [Confirms]
313-313	29	<u>Contributor2</u> : What? [Confirms]
313-318	29	<u>Contributor1</u> : Yeah, make it just, make it go under, like the black and under the ball. [Instructions]
... ..		

ined the temporal differences that involved “confirms understanding”: 1) “confirms understanding” ↔ “provides reasons to support a solution” in the “successful” model and 2) “confirms understanding” ↔ “provides instructional support” in the “unsuccessful” model. The link between people and game was reflected by “screen motion” involved connections. Among the three such relations (all in the “successful” model), the “screen motion” → “talks about challenge situations” was easy to understand and “screen motion” → “no indicator utterances” involved language that was not captured by our coding scheme. Thus, we examined only the “screen motion” → “asks others for suggestions”.

Table 5 shows two typical scenarios for the “screen motion”

→ “asks others for suggestions”, showing the time in the block in seconds (Time), order of the window in the level attempt sequence (Window), and the speech/actions that occurred in the window. In example 1, the controller asked for suggestions right after a level started, where a screen refresh resulted in a large screen motion. In example 2, the controller asked for suggestions after a failed attempt that contains a series of large in-game motions. These examples revealed that in successful level attempts, controllers were likely to ask for suggestions according to the current game situation.

Next, we investigate the bidirectional relation “confirms understanding” ↔ “provides reasons to support a solution”.

An examination of the coded sequences showed that in most cases, these two relations exist independent of each other, rather than form a semantically connected loop. So, we investigate each of them independently as shown in Table 6. In the “confirms understanding” \rightarrow “provides reasons to support a solution” example, a contributor answered the controller’s clarification question with an instruction and a reason to clarify its purpose (to catch the right timing, which is the key to solve the level). This suggests that in successful CPS processes, people were likely to answer clarification questions with reasons. In the “provides reasons to support a solution” \rightarrow “confirms understanding” example, a contributor proposed an idea to move the object, and gave a reason to clarify the goal, but the controller did not fully understand the idea and asked a clarification question. This relation does not align with our intuition that providing reasons increases clarity and reduces confusion. A potential explanation is that reasons were used conditionally on unclear or complex ideas, rather than unconditionally or randomly on all ideas. However, more investigations are needed to understand how reasons were provided.

An examination of the coded sequences for the bidirectional relation “confirms understanding” \leftrightarrow “provides instructional support” showed that there exist many cases where these two relations were semantically connected, forming a loop. Table 7 shows an example of it, where we can clearly see that Contributor1’s unclear instructions repeatedly confused both the other two people and raised clarification questions. Note that the examples in Table 6 and Table 7 indicated that semantic connections do exist between utterances that are not in consecutive order. This suggests that aggregating utterances with windows is a valid way to examine the temporal relations between actions.

6. DISCUSSION

This work applied multilevel vector autoregression (mlVAR) to investigate the temporal similarities and differences in verbal behaviors between successful vs. unsuccessful collaborative problem solving (CPS) outcomes. The remainder of this section discusses the main findings, applications, limitations and future work.

Main findings Our comparison of the “successful” vs. “unsuccessful” model together with a qualitative analysis revealed six between-action temporal relationships common to both, six unique to successful level attempts, and another eight unique to unsuccessful level attempts. The common relations suggest that the process of establishing shared understanding via asking and reacting to “confirms understanding” (clarification) questions was common to both successful and unsuccessful level attempts. For successful level attempts, people were likely to answer “confirms understanding” questions with reasons and to ask for suggestions according to the current game situation, while for unsuccessful level attempts, teams were more likely to struggle with unclear instructions and to respond to inappropriate ideas. Next, we discuss why certain behaviors in successful level attempts may facilitate CPS processes and what makes it hard to convey instructions.

Our analysis revealed that controllers in successful level attempts were likely to ask for suggestions according to the

current game situation (typically when they got stuck). This behavior may facilitate productive CPS because asking for suggestions can increase other people’s participation. Further, other people’s contributions are more helpful when the controller has no idea about how to solve the level. Additionally, in successful level attempts, people were more likely to answer clarification questions with reasons. As discussed above, this behavior may facilitate CPS because providing reasons enhances the clarity of the ideas/instructions being conveyed.

For unsuccessful level attempts, people were likely to struggle with unclear instructions. From the example in Table 7, we noticed at least two potential reasons for that. First, the team does not have a shared concept of a springboard, and thus Contributor1 had to describe how to make it (“connect the two dots, then like put it under the ball”) and how it works (“it’s going to end up falling on the bouncy board and hopefully go up”). Second, Contributor1 had to describe everything via words, rather than point to the screen directly, which largely reduced the clarity of her descriptions.

Future applications: A potential application of this work is to support the provision of timely CPS feedback in computer supported collaboration environments. For example, when a “confirms understanding” \leftrightarrow “provides instructional support” loop is detected, the system could send an interactive message to the team asking whether people are struggling with conveying clear instructions. If so, the system can provide suggestions such as describing the high level idea rather than low level actions. Another potential application is to provide CPS analytics for people to inspect their collaboration behaviors and improve CPS skills.

Limitations and future work: A limitation of our work is that we aggregated the data with 10-second windows, and thus the mlVAR analysis only effectively captures the connections between two consecutive windows. Connections with a short or longer temporal distance cannot be effectively discovered. Future studies could aggregate the data with different window sizes and see what other temporal patterns can be found. The second limitation is our data were collected in a specific lab setting where triads collaborated via video conferencing to play a physics learning game. Thus, it is unclear whether our findings can be generalized to other collaboration scenarios (e.g. real-world activities, face-to-face coordination). Third, we did not account for people’s roles in our analysis (due to limited sample size). Roles can largely influence people’s actions (e.g. only contributors “provide instructional support”), and therefore taking roles into account may provide more insights into the CPS processes. Finally, we did not compare mlVAR with other temporal analysis approaches (e.g. lag sequential analysis) to see whether they would generate similar or different insights. This is a potential future research direction.

7. ACKNOWLEDGMENTS

This research was supported by the NSF National AI Institute for Student-AI Teaming (iSAT) (DRL 2019805) and NSF DUE 1745442/1660877. The opinions expressed are those of the authors and do not represent views of the funding agencies.

8. REFERENCES

- [1] J. Andrews-Todd and C. M. Forsyth. Exploring social and cognitive dimensions of collaborative problem solving in an open online simulation-based task. *Computers in human behavior*, 104:105759, 2020.
- [2] B. Barron. Achieving coordination in collaborative problem-solving groups. *The journal of the learning sciences*, 9(4):403–436, 2000.
- [3] C. A. Bowers, C. C. Braun, and B. B. Morgan Jr. Team workload: Its meaning and measurement. In *Team performance assessment and measurement*, pages 97–120. Psychology Press, 1997.
- [4] L. F. Bringmann, N. Vissers, M. Wichers, N. Geschwind, P. Kuppens, F. Peeters, D. Borsboom, and F. Tuerlinckx. A network approach to psychopathology: new insights into clinical longitudinal data. *PLoS one*, 8(4):e60188, 2013.
- [5] E. Care, C. Scoular, and P. Griffin. Assessment of collaborative problem solving in education environments. *Applied Measurement in Education*, 29(4):250–264, 2016.
- [6] C.-J. Chang, M.-H. Chang, B.-C. Chiu, C.-C. Liu, S.-H. F. Chiang, C.-T. Wen, F.-K. Hwang, Y.-T. Wu, P.-Y. Chao, C.-H. Lai, et al. An analysis of student collaborative problem solving activities mediated by collaborative simulations. *Computers & Education*, 114:222–235, 2017.
- [7] A. Csanadi, B. Eagan, I. Kollar, D. W. Shaffer, and F. Fischer. When coding-and-counting is not enough: using epistemic network analysis (ena) to analyze verbal data in cscl research. *International Journal of Computer-Supported Collaborative Learning*, 13(4):419–438, 2018.
- [8] A. Deppermann and S. Günthner. *Temporality in interaction*, volume 27. John Benjamins Publishing Company, 2015.
- [9] N. M. Dowell, T. M. Nixon, and A. C. Graesser. Group communication analysis: A computational linguistics approach for detecting sociocognitive roles in multiparty interactions. *Behavior research methods*, 51(3):1007–1041, 2019.
- [10] O. V. Ebrahimi, J. Burger, A. Hoffart, and S. U. Johnson. Within-and across-day patterns of interplay between depressive symptoms and related psychopathological processes: a dynamic network approach during the covid-19 pandemic. *BMC medicine*, 19(1):1–17, 2021.
- [11] L. Eloy, A. EB Stewart, M. Jean Amon, C. Reinhardt, A. Michaels, C. Sun, V. Shute, N. D. Duran, and S. D’Mello. Modeling team-level multimodal dynamics during multiparty collaboration. In *2019 International Conference on Multimodal Interaction*, pages 244–258, 2019.
- [12] S. Epskamp, M. K. Deserno, and L. F. Bringmann. *mlVAR: Multi-Level Vector Autoregression*, 2021. R package version 0.5.
- [13] S. Epskamp, L. J. Waldorp, R. Möttus, and D. Borsboom. The Gaussian Graphical Model in Cross-Sectional and Time-Series Data. *Multivariate Behavioral Research*, 53(4):453–480, jul 2018.
- [14] D. A. Ferrucci. Introduction to “this is watson”. *IBM Journal of Research and Development*, 56(3.4):1–1, 2012.
- [15] K. L. Gwet. Computing inter-rater reliability and its variance in the presence of high agreement. *British Journal of Mathematical and Statistical Psychology*, 61(1):29–48, 2008.
- [16] L. M. Hamel, R. Moulder, F. W. Harper, L. A. Penner, T. L. Albrecht, and S. Eggly. Examining the dynamic nature of nonverbal communication between black patients with cancer and their oncologists. *Cancer*, 127(7):1080–1090, 2021.
- [17] J. Hao, L. Liu, A. A. von Davier, and P. C. Kyllonen. Initial steps towards a standardized assessment for collaborative problem solving (cps): Practical challenges and strategies. In *Innovative assessment of collaboration*, pages 135–156. Springer, 2017.
- [18] F. Hesse, E. Care, J. Buder, K. Sassenberg, and P. Griffin. A framework for teachable collaborative problem solving skills. In *Assessment and teaching of 21st century skills*, pages 37–56. Springer, 2015.
- [19] M. Kapur. Temporality matters: Advancing a method for analyzing problem-solving processes in a computer-supported collaborative environment. *International Journal of Computer-Supported Collaborative Learning*, 6(1):39–56, 2011.
- [20] R. Kievit, W. E. Frankenhuis, L. Waldorp, and D. Borsboom. Simpson’s paradox in psychological science: a practical guide. *Frontiers in psychology*, 4:513, 2013.
- [21] M. Laal. Positive interdependence in collaborative learning. *Procedia-Social and Behavioral Sciences*, 93:1433–1437, 2013.
- [22] R. M. Maldonado, K. Yacef, J. Kay, A. Kharrufa, and A. Al-Qaraghuli. Analysing frequent sequential patterns of collaborative learning activity around an interactive tabletop. In *Educational Data Mining 2011*, 2010.
- [23] M. A. Marks, J. E. Mathieu, and S. J. Zaccaro. A temporally based framework and taxonomy of team processes. *Academy of management review*, 26(3):356–376, 2001.
- [24] C. Marquart, Z. Swiecki, B. Eagan, and D. Shaffer. *ncoder*, 2018.
- [25] S. Nebel, S. Schneider, M. Beege, F. Kolda, V. Mackiewicz, and G. D. Rey. You cannot do this alone! increasing task interdependence in cooperative educational videogames to encourage collaboration. *Educational Technology Research and Development*, 65(4):993–1014, 2017.
- [26] OECD. Pisa 2015 collaborative problem-solving framework. organisation for economic co-operation and development. page 131–188, 2017.
- [27] T. Okada and H. A. Simon. Collaborative discovery in a scientific domain. *Cognitive science*, 21(2):109–146, 1997.
- [28] J. K. Olsen, D. M. Belenky, V. Alevan, and N. Rummel. Using an intelligent tutoring system to support collaborative as well as individual learning. In *International conference on intelligent tutoring systems*, pages 134–143. Springer, 2014.
- [29] C. O’Malley, S. Langton, A. Anderson, G. Doherty-Sneddon, and V. Bruce. Comparison of face-to-face and video-mediated interaction.

- Interacting with computers*, 8(2):177–192, 1996.
- [30] D. Perera, J. Kay, I. Koprinska, K. Yacef, and O. R. Zaïane. Clustering and sequential pattern mining of online collaborative learning data. *IEEE Transactions on Knowledge and Data Engineering*, 21(6):759–772, 2008.
- [31] M. S. Poole and M. E. Holmes. Decision development in computer-assisted group decision making. *Human Communication Research*, 22(1):90–127, 1995.
- [32] J. Roschelle and S. D. Teasley. The construction of shared knowledge in collaborative problem solving. In *Computer supported collaborative learning*, pages 69–97. Springer, 1995.
- [33] Y. Rosen. Assessing students in human-to-agent settings to inform collaborative problem-solving learning. *Journal of Educational Measurement*, 54(1):36–53, 2017.
- [34] Y. Rosen, I. Wolf, and K. Stoeffler. Fostering collaborative problem solving skills in science: The animalia project. *Computers in Human Behavior*, 104:105922, 2020.
- [35] N. Rummel and H. Spada. Learning to collaborate: An instructional approach to promoting collaborative problem solving in computer-mediated settings. *The journal of the Learning Sciences*, 14(2):201–241, 2005.
- [36] C. Scoular and E. Care. Monitoring patterns of social and cognitive student behaviors in online collaborative problem solving assessments. *Computers in Human Behavior*, 104:105874, 2020.
- [37] V. Shute, R. Almond, and S. Rahimi. Physics playground (1.3)[computer software]. <https://pluto.coe.fsu.edu/ppteam/pp-links/>, 2019.
- [38] C. Sun, V. J. Shute, A. Stewart, J. Yonehiro, N. Duran, and S. D’Mello. Towards a generalized competency model of collaborative problem solving. *Computers & Education*, 143:103672, 2020.
- [39] C. Sun, V. J. Shute, A. E. Stewart, Q. Beck-White, C. R. Reinhardt, G. Zhou, N. Duran, and S. K. D’Mello. The relationship between collaborative problem solving behaviors and solution outcomes in a game-based learning environment. *Computers in Human Behavior*, 128:107120, 2022.
- [40] D. Suthers, L. Girardeau, and C. Hundhausen. Deictic roles of external representations in face-to-face and online collaboration. In *Designing for change in networked learning environments*, pages 173–182. Springer, 2003.
- [41] Z. Swiecki, A. R. Ruis, C. Farrell, and D. W. Shaffer. Assessing individual contributions to collaborative problem solving: a network analysis approach. *Computers in Human Behavior*, 104:105876, 2020.
- [42] Y. R. Tausczik, A. Kittur, and R. E. Kraut. Collaborative problem solving: A study of mathoverflow. In *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing*, pages 355–367, 2014.
- [43] H. Vrzakova, M. J. Amon, M. Rees, M. Faber, and S. D’Mello. Looking for a deal? visual social attention during negotiations via mixed media videoconferencing. *Proceedings of the ACM on Human-Computer Interaction*, 4(CSCW3):1–35, 2021.
- [44] H. Vrzakova, M. J. Amon, A. Stewart, N. D. Duran, and S. K. D’Mello. Focused or stuck together: multimodal patterns reveal triads’ performance in collaborative problem solving. In *Proceedings of the tenth international conference on learning analytics & knowledge*, pages 295–304, 2020.
- [45] J. K. Westlund, S. K. D’Mello, and A. M. Olney. Motion tracker: Camera-based monitoring of bodily movements using motion silhouettes. *PLoS one*, 10(6):e0130293, 2015.
- [46] J. R. Zubizarreta, M. J. R. Zubizarreta, and G. SystemRequirements. Package ‘designmatch’. *Matched Samples that are Balanced and Representative by Design Version 0.3. 0*, 2018.

Challenges and Feasibility of Automatic Speech Recognition for Modeling Student Collaborative Discourse in Classrooms

Rosy Southwell, Samuel Pugh, E. Margaret Perkoff, Charis Clevenger, Jeffrey B. Bush, Rachel Lieber, Wayne Ward, Peter Foltz, Sidney D’Mello

Institute of Cognitive Science
University of Colorado Boulder
Boulder, CO 80309
roso8920@colorado.edu

ABSTRACT

Automatic speech recognition (ASR) has considerable potential to model aspects of classroom discourse with the goals of automated assessment, feedback, and instructional support. However, modeling student talk is besieged by numerous challenges including a lack of data for child speech, low signal to noise ratio, speech disfluencies, and multiparty chatter. This raises the question as to whether contemporary ASR systems, which are benchmarked on adult speech in idealized conditions, can be used to transcribe child speech in classroom settings. To address this question, we collected a dataset of 32 audio recordings of 30 middle-school students engaged in small group work (dyads, triads and tetrads) in authentic classroom settings. The audio was sampled, segmented, and transcribed by humans as well as three ASR engines (Google, Rev.ai, IBM Watson). Whereas all three ASRs had high word error rates, these mainly consisted of deletion errors. Further, Google successfully transcribed a greater proportion of utterances than the other two, but with more word substitutions; insertions were low across the board. ASR accuracy was robust to different speakers and recording idiosyncrasies evidenced by <5% of variance in error rates attributable to the student and recording session. We found that ASR errors had a larger negative effect on downstream natural language processing tasks at the word, phrase, and semantic levels rather than at the discourse level. Our findings indicate that ASR can be used to extract meaningful information from noisy classroom speech and might be more suitable for applications that require higher precision but are tolerant of lower recall.

Keywords

automatic speech recognition, collaborative problem solving, classroom speech, natural language understanding

1. INTRODUCTION

Students learn by telling and doing. Indeed, decades of educational research has converged on one (among several) perspectives of learning as a social and collaborative activity [8, 89, 97]. Effective collaborative learning (CL) activities give students the opportunity to work together towards a common goal, share their ideas and

build upon the ideas of others, negotiate strategies, monitor execution of plans, and reflect on outcomes [17, 28, 33, 39, 72, 75]. Thus, the benefits of CL are manifested not only in the acquisition of domain knowledge [86], but also in the development of essential 21st century skills such as collaborative problem solving and critical thinking [27, 29].

Despite a strong consensus on the value and merits of CL, its widespread implementation in contemporary classrooms is limited. A key factor limiting its adoption is that it is extremely challenging for teachers to effectively orchestrate rich CL activities in their classes. To support successful CL, teachers must monitor group progress on time-sensitive activities, provide guidance and help when students get stuck and risk disengagement, and ensure that students engage in productive knowledge-building conversations, all while ensuring that classroom norms for respectful discourse are maintained [71, 88]. To complicate things further, teachers must perform these demanding activities simultaneously across multiple (often 5-10) groups – a daunting assignment. Can intelligent systems, which unlike teachers, are able to be omnipresent across multiple student groups, enhance teachers’ ability to scaffold rich CL experiences for all their students?

One exciting possibility is to design systems capable of natural language understanding (NLU) to support CL in student groups. Indeed, the linguistic content of discourse during CL is considered the “gold mine of information” on how students acquire knowledge and skills [32, 73]. However, despite an extensive body of research demonstrating the utility of other modalities (e.g., body movement, gesture, eye-gaze, paralinguistics, see review [62] for automatically analyzing collaboration, an automated approach for capturing, transcribing, and analyzing student speech during face-to-face CL in the classroom has yet to be developed. Most language-based approaches to date thereby rely on typed transcripts from chats (or human-transcribed speech) to analyze and support collaborative discourse [21, 30, 52, 76].

At the heart of this challenge lies an extremely difficult technical hurdle: using automatic speech recognition (ASR) to obtain accurate (or even serviceable) transcriptions of student discourse in noisy, real-world classrooms. This endeavor is complicated by multiple compounding challenges. Namely, with upwards of 20-30 students in a typical US classroom [57] with multiple student groups simultaneously engaged in CL activities, speech signals are obfuscated by background chatter and ambient noise. In addition, ASR systems already have difficulty recognizing children’s speech (even in ideal, noise-free environments), as they tend to speak less clearly than adults [46]. In fact, even the basic acoustic

R. Southwell, S. Pugh, E. M. Perkoff, C. Clevenger, J. Bush, R. Lieber, W. Ward, P. Foltz, and S. D’Mello. Challenges and feasibility of automatic speech recognition for modeling student collaborative discourse in classrooms. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 302–315, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853109>

characteristics of children’s voices and language use [25, 46], differ from adults (on whose voices most ASR systems are trained), resulting in a degradation in performance when these systems are applied to children’s speech [70]. Multiparty speech recognition is another challenge for ASR [12, 65], where utterances - from an unknown number of unique speakers - may overlap, whereas ASR systems are generally trained on audio where speakers have already been separated.

Despite these challenges, pursuing technologies capable of automatically capturing and analyzing student speech during face-to-face CL in authentic school environments is an important avenue of research. These technologies have the potential to significantly improve orchestration and support of CL [73], whether by providing teachers with feedback on progress of student groups (e.g., via a teacher dashboard [87]), or enabling real-time interventions to guide groups of learners towards equitable and productive collaboration.

In this paper we take a first step towards understanding the feasibility and challenges of automatically analyzing student speech in classrooms. Specifically, we investigate: (1) patterns of errors in widely used commercial ASR systems for transcribing student discourse in authentic collaborative learning settings; and (2) the influence of ASR errors on downstream natural language understanding tasks at the word, phrase, semantic, and discourse levels. In doing so, we take an important step towards deploying speech-based collaborative learning technologies in classrooms.

1.1 Background and Related works

There is a large body of research on analyzing student- and teacher-classroom discourse [10, 54], so to keep scope manageable we focus on the automatic analysis of student speech and classroom speech.

1.1.1 Challenges with child speech recognition

Speech recognition in children is a well-documented challenge, with recognition accuracy substantially lower for children’s speech than adult’s [25, 61, 74]. Yet, both commercially-available and research ASR systems are generally trained with clean audio data from adult speakers, with one speaker per utterance, and often reading from a script, which perform substantially worse on realistic, spontaneous speech [83]. These systems do not easily generalize to child speech where vocal characteristics such as higher fundamental and formant frequency and greater variability in pitch, and linguistic factors such as disfluency rate, differ between children and adults, as well as changing as children mature [25, 46]. In an analysis of Google, Bing and Nuance ASR systems, [70] found that age significantly impacted performance for all ASRs except Google. Further, accented speech of non-native speakers impacts ASR performance, as articulation and pronunciation differ from the training data [19]. The classroom setting provides an additional challenge. Howard et al. [34] reported that the typical classroom signal-to-noise ratios range from -7 dB to +5 dB, further impeding ASR [95]. Finally, microphone placement impacts recognition - the further the speaker from the microphone, the greater the impact of reverberation and other signal degradation on ASR [23, 56].

1.1.2 Child speech recognition in controlled learning environments

Numerous educational applications which use ASR on children’s speech have been developed, albeit outside of the hustle and bustle of the classroom. One strand of research uses ASR as part of automatic reading tutors for young children learning to read aloud

from text. Here, the reference (ground-truth) transcript is available in the form of reading materials, and by comparing this to the ASR output, pronunciation errors can be identified and fed back to the student or their teacher [3, 51, 60, 66]. Generally, these systems are used in a quiet environment such as a library [66], and in all cases are designed with the expectation that only a single speaker is reading at a time. Another application of ASR is in conversational tutors, where both speech recognition and language generation are combined in a real-time system. One example is My Science Tutor (MyST [92]) which supports one-on-one and small-group science learning [13]. The MyST ASR system was trained using a dataset of elementary school students, and achieves a word error rate (WER) of 0.30 (about 70% accuracy) on a reduced vocabulary of ~6000 words. However tutoring sessions did not take place in the main classroom, and users wore a headset, both of which avoided some of the key challenges of classroom ASR. Online learning environments also simplify the collection of clean, speaker-separated speech recordings, and several examples exist of automated analysis of student-teacher dialog starting from ASR transcripts [47, 94].

1.1.3 Automated analysis of teacher speech in the classroom

Recent advances in ASR make the prospect of sufficiently accurate transcription of speech in the classroom a possibility. Most of the ASR literature focuses on adult speech, and this is mirrored in the availability of commercially available, cloud-based ASR APIs, (for examples, see [20] but see [15] for a child-tailored ASR service). As a result, most automated approaches have focused on analyzing teacher speech with varying degrees of automation including ASR-only [9, 37, 38, 42, 81, 96], human transcripts [82], or a combination of both [7]. There are also differences in the depth of the construct being modeled. For example, Zylich and Whitehill [96] recently aimed to automatically detect 21 key phrases (e.g., “good job”) in teacher talk from audio, but stopped short of measuring pertinent discourse constructs. In contrast, Kelly et al. [42] and Jensen et al. [37, 38] developed fully automated approaches to model five features of discourse: questions (vs. statements), authentic (open-ended), instructional utterances, elaborated evaluations, cognitive level, goal specificity, and presence of disciplinary terms.

One advantage of focusing on teachers is that it is easier to affix high-quality microphones on a single teacher than an entire classroom of students. For example, the Kelly and Jensen studies used a unidirectional, noise-canceling microphone with cardioid pickup pattern which is most sensitive to sounds from the front of the mic, thereby canceling background noise [37, 38, 42]. Despite a high-quality mic, classroom ASR is still challenging due to background noise, multidisciplinary chatter, dialectical variations, and so on. To this point, [5] and [18] compared several ASR engines for accuracy in transcribing teacher speech recorded in authentic classrooms. These two studies tested 7 ASRs yielding word error rates ranging from .31 to 1.00.

It is important that these studies are replicated due to the rapid advancement in ASR technologies each year. For example, using the same microphone and ASR engine on similar classroom data, Jensen et al. [37] obtained a major reduction in error (from 44% WER to 28%) in 2020 compared to Blanchard’s (2015) study [5].

1.1.4 Automated analysis of student speech in classrooms

Examples of automated analysis of classroom audio focused on student speech are rare, as justified by the many acoustic and

linguistic challenges inherent in the full pipeline from recording speech, to transcription in the context of overlapping, non-adult speakers in a noisy environment, to extracting meaning from language patterns of students still undergoing linguistic development. Nevertheless, several recent works have utilized non-specialized commercial ASR services for child speech with promising results - demonstrating that ASR transcriptions can be used to derive useful downstream measures despite very high WER [64, 84].

To our knowledge, the only example where ASR is used to transcribe conversations among students as input to an NLP model is in the context of a collaborative problem solving (CPS) study conducted in both the classroom and lab [64]. Here, students aged 12-15 participated in two CPS activities in math and physics. Participants wore headsets with microphones and completed the task (in dyads) over Zoom from a shared computer lab at the school, or for a subset of participants, in a laboratory. Captured speech was manually segmented into utterances, then transcribed using the IBM Watson speech-to-text service [36]. Performance degradation attributable to the classroom environment was evident, with a word error rate (WER) of 0.78 in the classroom, meaning only 22% of human-transcribed words were correctly transcribed, as compared to a WER of 0.54 for dyads recorded in the laboratory.

ASR has also been used to capture classroom conversation in preschool children. Lileikyte et al. [48] used LENA’s wearable audio recorders, which are designed for capturing speech in young children, to train an ASR with custom acoustic and language models using data augmentation, obtaining a WER of 0.64 on spontaneous conversation in 2–5-year-old children. Using the same wearable devices in preschoolers, Tao et al. [84] ran audio through Google Cloud ASR [26] and used the transcripts to derive network representations of groups in social interactions based on word count vector similarity between utterances, though ASR accuracy is not reported. Further, the use of LENA is cost-prohibitive, with pricing in the thousands of dollars, which is infeasible at scale.

Beyond these examples, speech analysis in the classroom is limited to extraction of non-linguistic (i.e., acoustic/prosodic) features, which nevertheless show promise for classification of discourse categories [6, 40, 91], speaker identification [84] and diarization to identify speaker turns [49, 53].

1.1.5 *Is perfect ASR needed?*

As reviewed above, ASR in the classroom is beset by many challenges, especially for analyzing student speech. However, the goal of many applications is not to obtain perfect transcripts of speech, but to use the transcripts for downstream NLU tasks relevant to education (e.g., assessment, feedback, intervention). Indeed previous research has indicated that useful information can be obtained from imperfect transcripts. Pugh et al. [64] found that using ASR instead of human transcripts led to only a 14% decrease in classifier performance (still significantly above chance) despite a WER of 0.78. Outside the classroom, Stewart et al. [78] reported a mere 4.2% decrease in accuracy for classifying collaborative skills using ASR versus human transcripts. Indeed, the question of robustness of models of team performance to simulated ASR errors was addressed by [22], with even a WER of 57% only decreasing classifier performance by 20% relative to perfect transcription. The authors suggest that the constrained, contextualized nature of conversation makes discourse-level NLP models robust to modifications of individual words.

Of course, there is likely an upper limit to errors beyond which the signal to noise ratio is too low to be useful, a likely possibility for

analyzing multiparty collaborative child speech in the classroom. This raises the questions of whether it is feasible to obtain meaningful information on student collaborative discourse despite noisy ASR and to what extent do ASR errors impact the meaning conveyed in an utterance and how does this impact downstream NLU tasks.

1.2 **Current Study, Contribution, & Novelty**

In this study, we take an important first step towards the automated analysis of student collaborative discourse in noisy, authentic classrooms. We compare a variety of commercially available ASR systems on both speech to text transcription, and we investigate the influence of ASR errors on downstream NLU tasks using a novel dataset of audio recordings from real-world middle-school classrooms where multiple student groups are engaged in CL.

Specifically, we quantify ASR performance in terms of traditional evaluation metrics (e.g., Word Error Rate [WER]), and investigate the types of speech recognition errors encountered (e.g., substitutions, deletions). Further, we seek an understanding of the sources of variability in ASR errors at the level of the utterance, student, and session by systematically sampling students across multiple recording contexts (i.e., across different lessons, student groups, and days). This information can provide insights into potential disparities of ASR systems, which may have unequal impacts on individual student outcomes when used as inputs to downstream applications. To this point, we also compare ASR errors and their influence on downstream NLU applications (e.g., semantic similarity of transcripts [43], recognition of task-relevant content words, assessing collaboration skills) to probe the feasibility of using automated transcripts for NLU-based CL analytics in the classroom.

To our knowledge, this is the first attempt to systematically analyze automated transcriptions of face-to-face student collaborative discourse in a real K-12 school environment. Although other studies use ASR as input to language-based models of classroom discourse, the majority of these focus on teacher speech [5, 9, 14, 37, 38] or collaborative problem solving in adult undergraduates [63, 78, 79]. We also use inexpensive, commercially available microphones placed on the tabletop, each capturing speech from 2-4 students, which allows us to expose the challenges of capturing real-world classroom audio where multiple speakers are intermixed in a single-channel recording with additional impacts of reverberation and background noise. This contrasts with prior studies analyzing classroom audio, which mostly employ individual microphones to isolate speech [5, 37, 47, 48, 64]. Also, we use data collected in the context of a live, face-to-face discourse rather than an online learning environment [47, 94]. The choice to use table-top mics rather than individual noise-canceling lapel microphones or headsets is motivated both by practicality and cost considerations, and by the concern that individually miking students would feel intrusive and even impede collaboration.

Finally, with respect to scope, we focus on widely available commercial ASR services in lieu of customized ASR systems with acoustic and language models trained on our target demographic and data. This may disadvantage speech recognition performance, however using publicly available ASR providers is desirable for practical reasons including the simplicity of integration due to a well-documented API, and the likely continuation of updates to the model in the future. We also don’t seek to improve or engineer better performance out of these systems in the current work because the goal is to establish baseline performance of out-the-box ASR

systems on the difficult task of analyzing child collaborative talk in noisy classrooms.

2. METHODS

2.1 Data Collection

The data was collected as part of a larger project involving a Research-Practice Partnership [41] focused on using co-design and professional learning to support the use of programmable sensor technology and computational thinking for authentic inquiry in middle school science and STEM classrooms [4]. We analyzed audio and video data from one participating U.S. public middle school teacher in this work.

2.1.1 Learning Context: Sensor Immersion

Participating teachers implemented a multi-day curriculum unit called Sensor Immersion that focuses on students working collaboratively to understand how to program and wire sensors to collect data about their local environments, empowering students to be data producers [31] and answer questions that they find personally meaningful and relevant. The Sensor Immersion curriculum uses an interactive data display called the Data Sensor Hub (DaSH [11]; Figure 1) as an anchoring phenomenon [24]. Students explore the system, create scientific models and learn to replicate its functionality in the context of their own investigations. Along the way, students develop a program that can control a variety of physical sensors including a sound sensor, moisture sensor, and an environmental sensor.

Sensor Immersion is broken down into five lessons, each of which can span multiple days. Lesson 1 focuses on question generation and modeling. Throughout the following lessons, students work to answer their questions about how the DaSH works. To do so they learn to program and wire the sensors, working in pairs doing a pair-programming task using MakeCode block programming (Figure 2). Students gradually build on their understanding of programming and sensors by working together to program and wire one sensor and eventually building and programming a sensor

system to answer questions about a personally meaningful phenomenon. Opportunities for small-group collaboration around these sensors and their programming are designed into each lesson.

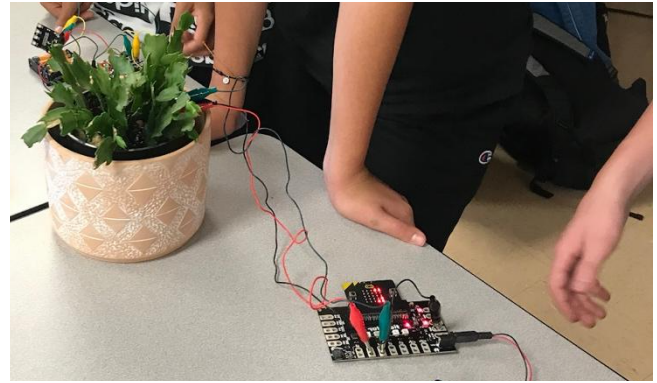


Figure 1. Close-up of the DaSH system which links sensors to the computing interface. Various sensors can be wired to the system to measure local environmental conditions such as soil moisture levels (pictured), CO2, humidity, temperature and ambient room noise.

2.1.2 Participants

The data sample included 30 students from 4 cohorts taught by a single teacher in a suburban school district in the US. All procedures were approved by designated Institutional Research Boards and data were only collected from students who provided both personal assent and their parent’s signed consent forms. Most of the students were in the 6th-8th grades except for one class of 5th graders. Across the school district, the ethnicity of students enrolled (as of the 2021-2022 school year) was as follows: 62% White, 30% Hispanic, 3% Asian, 3% two or more races, 1% Black, 0.3% American Indian or Alaska Native, and 0.1% Hawaiian/Pacific Islander [77]. About half (49%) were female.

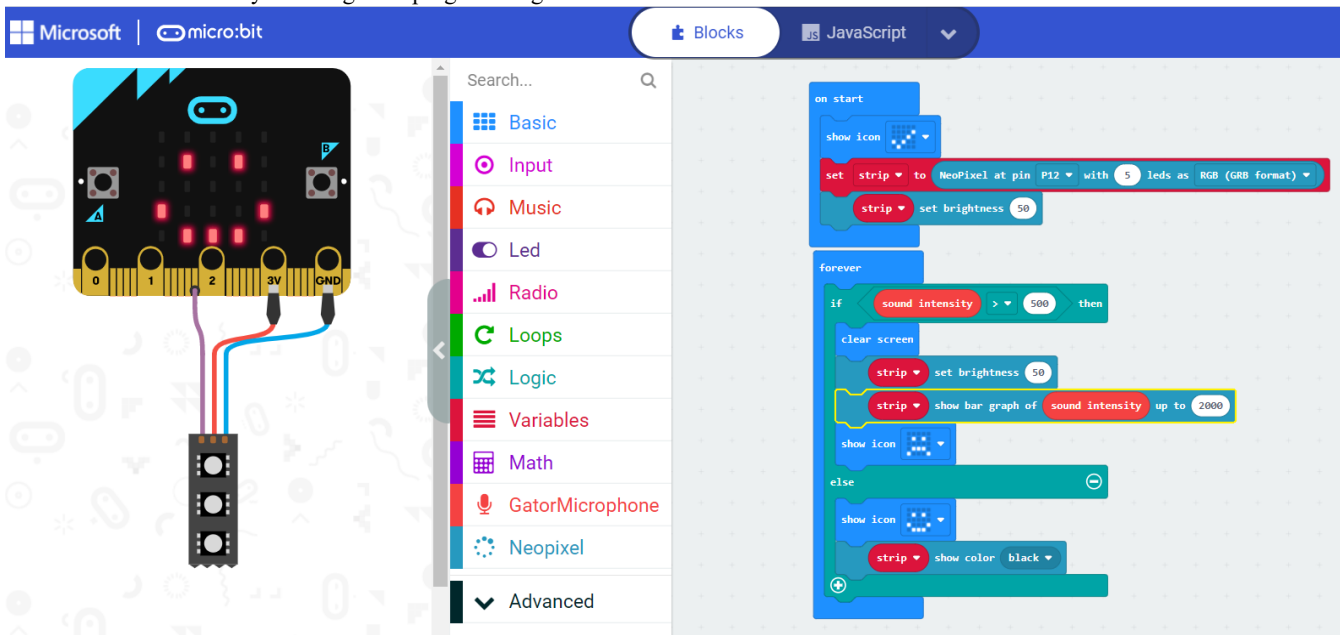


Figure 2. Screenshot of the MakeCode programming interface

2.1.3 Microphone

Our choice of microphone was influenced by several factors including audio quality, cost, power source, form-factor, and ease of use. We evaluated a range of candidates (e.g., MXL, Sony ICD PX370, ZOOM H1n, AudioTechnica-ATR, AudioTechnica-Omni, AudioTechnica-Cardioid, ProCon, Saramonic), we chose the Yeti Blue. This microphone has a user-configurable effective pickup pattern: omnidirectional, polar, XY stereo or cardioid, costing around \$100USD. It is USB-powered, enabling use with an iPad without the need for an external mixer or phantom power.

2.1.4 Procedure

During each class, researchers placed microphones (Yeti Blue) at tables around the classroom. Groups of 2-4 consenting students were seated at each table. Depending on the lesson the students either worked as a team or as multiple dyads (during pair programming). The microphone was placed on the table roughly equidistant from all students, using the omnidirectional setting when recording 3 or more students, or the stereo setting when two students were sitting at either side of the microphone. The microphone was connected to an iPad via USB that hosted the recording software recording at 48kHz sampling rate. We also collected iPad video from a subset of students. Microphones were set up by a researcher who recorded field notes on different events (e.g., start of lesson, start of small group work, technical failures).

2.2 Data Treatment

2.2.1 Sample Selection.

We opted to select recordings with both audio and video to aid in ground-truth speaker diarization efforts (i.e., who is speaking). Of a total of 118 recordings, 79 met this criterion, of which we selected 49 recordings which contained small-group work, where at least one student in each group appeared in a minimum of 4 recordings.

From each video, five 5-minute candidate samples were selected from within the small-group work segment of the lesson, constrained to the middle of the segment such that the random sample included the midpoint of the small-group interval as beginning and end of the task tended to include less on-topic speech. A researcher then listened to each of the five random samples in turn. If the sample met the 20-word criterion, then it was selected for the sample. If it did not meet the criteria ($n = 17$), the next segment was listened to and so forth. If none of the 5 segments met the criteria, then the recording was excluded entirely. Through this process, we ended with 32 samples totaling 160 minutes of speech from 30 students. A majority (70%) of the students were in at least two recordings (Table 1).

Table 1. Sample summary

	M (SD)	Range
No. students per recording	2.6 (0.7)	2-4
No. recordings per student	2.7 (1.6)	1-5
No. utterances per recording	61.6 (29.6)	21-139
No. utterances per student	65.7 (47.2)	10-188
Wordcount per utterance	4.55 (4.03)	1-47

2.2.2 Transcription and annotation

Samples were transcribed in ELAN annotation software by trained transcribers, who recorded millisecond-resolution timestamps (based on the audio waveform) for utterance start and end times along with speaker identity. Where speaker identity was clear, but speech was too indistinct to transcribe, some or all of the utterance content was coded as "[inaudible]". This resulted in 2207 student

utterances, of which 1970 contained at least 1 audible word (See Table 1).

Utterance-level audio segments were automatically transcribed by three cloud-based ASR services: Google Speech-to-text [26], Rev.ai [68], and IBM Watson [36]. We selected Google because it has been shown to work as well for children as adults [70] and in a recent review was shown to outperform similar services [20]. Watson has been used in multiple publications for ASR transcription of teacher talk [5, 37, 38] and as input to CPS linguistic models [63, 64]. Rev.ai was used as they claim equal or greater performance than Google [69]. We deemed these three ASRs sufficient for the present purposes of investigating patterns in and downstream influences of ASR errors and not to evaluate all available commercial ASR engines.

For Google, audio was first segmented using the human-segmented utterance-boundaries and individually submitted to the ASR. We used the video-optimized model as this was determined to outperform the default model in preliminary testing. For Rev.ai and Watson, all utterances from a given recording were concatenated before transcribing, as this theoretically allows the models to use prior language context to boost performance. The ASR result contains word-level timestamps which were used to split the full transcript back into the original utterances. We also tested using per-utterance transcripts for Watson and the Google streaming speech recognition API using the *single_utterance=True* option optimized for short utterances. Due to poorer performance than the main Watson and Google models, these were not analyzed further.

2.3 Measures

Before computing measures on the transcripts, all texts (human and ASR transcribed) were normalized to facilitate comparison. Non-word indicators used by the transcribers and ASR systems such as "[inaudible]", "[redacted]" and "%HESITATION" were stripped out. Numbers were spelled out if transcribed as digits. Leading and trailing punctuation was stripped from each word, and hyphens replaced by space. Finally, all words were transposed to lowercase.

2.3.1 Word Error Rate.

Using standard procedures [83], for each utterance, we used the Levenshtein algorithm at the word-level, which finds the minimum number of word substitution (S), insertion (I) and deletion (D) operations to align the reference (human transcript) to the hypothesis (ASR transcript). We used word error rate (WER) as a measure of transcription accuracy, which is given by: $WER = (S + D + I) / N_{reference}$ (number of words in the reference text). Proportion of insertion, substitution, and deletion errors were computed by dividing utterance-level error counts with the number of words in the human transcript. We also computed the number of words in the ASR transcripts along with a binary variable indicating whether the ASR returned any transcript at all.

2.3.2 Downstream NLP Measures

We focused on NLP tasks at the word, semantic, and discourse levels. In each case, we are interested in the error (distance) between the ASR (hypothesized) and human (reference) values.

2.3.2.1 BLEU Scores

The BLEU metric was developed to assess the performance of machine translation systems by comparing a gold standard translation to an output translation [59]. BLEU scores quantify sentence similarity based on modified n-gram precision, where scores vary from 0 (no match) to 1 (perfect match). This captures higher-order structure than WER: it is invariant to n-gram order,

and encapsulates longer subsequences than WER which is defined only at the individual word level. We computed the BLEU score for unigrams, bigrams, trigrams, and quad-grams and computed an unweighted average of the four, which was reversed (i.e., 1-BLEU) to get the BLEU distance (or error)

2.3.2.2 Topic Word Analysis.

At the word level, we quantified students’ uses of topic words that might be indicative of their cognitive engagement with the sensor immersion unit. The curriculum materials consist of storyboards, lesson plans, tutorials, etc., from which we generated a frequency dictionary of the named entities using the Named Entity Recognition algorithm from the Stanford CoreNLP toolkit [55]. Functional words were removed, resulting in 2,438 candidate words. Next, we used an existing Latent Dirichlet Allocation (LDA) topic model (created for an auxiliary purpose), which learns distinct topics from the document and returns the top 20 words that have the highest correlation with each of 20 topics. We computed the intersection of the 400 topic words and the 218 candidate named entities that occurred more than 20 times. This threshold ensured candidate words appeared at multiple points in the curriculum documents while keeping the list to a manageable size. This produced a set of 66 initial topic words. These topic words were then reviewed by curriculum experts who selected a subset of 33 topic words aligned to the following categories: science (e.g. *environmental*), coding (e.g. *function*), and wiring (e.g. *sensors*). For each utterance, we computed the number of topic words recognized by each ASR and the human transcript. To measure ASR fidelity specific to topic words, we compute *Topic Distance* as the absolute difference in utterance-level topic word counts between the human and ASR derived transcripts, with a lower bound of 0 and an undefined upper bound.

2.3.2.3 Semantic Distance

Beyond words themselves, we also evaluated the ASR transcripts using the semantic distance metric, which measures the similarity of a reference and a hypothesis transcript in a sentence-level

embedding space (using a pre-trained language model to obtain the embeddings), and has been shown to be a better predictor of performance on downstream NLP tasks than traditional metrics such as WER [43]. Following the procedure outlined in [43], we first extracted utterance-level embeddings using the *sentence-transformers* Python library [67] and the ‘*all-distilroberta-v1*’ model [50]. Then, we computed the cosine distances between the embeddings of each ASR (hypothesized) transcript and the reference human transcript. The cosine distance is defined as 1-cosine similarity (which ranges from -1 to 1), so it can take on values from 0 (identical) to 2 (dissimilar). To obtain a baseline value, we randomly shuffled the human transcripts within each 5-minute recording, then computed the semantic distance to each ASR transcript as described above. The average semantic distance over all ASRs was used as a baseline.

2.3.2.4 CPS Skill Classification

At the discourse level, we evaluated the utility of our ASR transcripts for a concrete NLP application: classifying collaborative problem solving (CPS) skills from student transcripts, which is one of the target applications noted in Section X. Specifically, we applied an existing classifier [63], which was trained to identify the following three CPS skills based on a validated CPS framework [80]: constructing shared knowledge; negotiation/coordination; maintaining team function, to our dataset. The classifier was a pre-trained BERT [16] model fine-tuned on a data set of 31,533 expert-coded student utterances (transcribed using the Watson ASR). Although the classifier was trained on a different dataset, it has been shown to be generalizable across domains [63], so we deemed it suitable for the present purposes. As such, we submitted both the human and ASR transcripts to the classifier, which outputs the predicted probabilities for the three CPS facets on each utterance. For each ASR, we computed the three-dimensional Euclidean distance between the ASR- and human- (reference) predicted probabilities as a measure of dissimilarity (CPS Distance). We also obtained a baseline shuffled value similar to the baseline Semantic Distance.

Table 2. Sample sentences and their corresponding ASR transcriptions. CPS codes: Const. = constructing shared knowledge; Neg. = negotiation/coordination; Maintain. = maintaining team function

Speaker	Human Transcript	Google	Watson	Rev	CPS Code
A	just start with the show number	start remove the show number	system started with the show numbers	start with the show number	Const.
B	oh	-	okay	okay	None
A	okay so you get rid of the show number	okc get rid of the sheriff	okay	okay so you get rid of the sharon remember	Maintain.
A	just drag it	stretch	dr don't	-	Maintain.
C	don't don't do that	don't don't do that	don't don't don't do that	don't do that	Maintain.
A	get rid of it	-	-	get rid of it	Maintain.
C	just okay	just	okay	it	Neg.
B	and now put this in this thing	i am for this and this thing	okay but this in this thing yeah	put this and this thing	Const.
A	yes	-	yeah	-	Neg.
C	no now you eat a taco	you know how you eat a taco	yeah are you talking	you need to talk	Maintain.
B	no do i put it in there	-	-	-	Const.
C	yeah	-	-	-	Neg.
A	yes	-	-	-	Neg.

2.4 Data Treatment

All measures (proportions of insertion, substitution and deletion errors; BLEU distance, Topic distance, semantic distance, CPS distance) were averaged per speaker per recording, resulting in 82 observations per ASR. This was done to obtain more reliable estimates due to the principle of aggregation. Because the distance metrics are only meaningful for utterances where the ASR returns a nonempty transcript, the averages for BLEU, Topic, Semantic and CPS distances were computed over nonempty transcripts only. To analyze the effects of ASR service and word errors on downstream measures, we used mixed effects linear regression models with speaker and recording identifier as random intercepts to account for the nested and repeated structure of the data with multiple speakers nested within recordings. Further, we used the *robustlmm* package in R [45], which provides estimates that are robust to outliers and other contaminants in the data. We used estimated marginal means (*emmeans* package in R) for pairwise comparisons using false-discovery rate adjustments for multiple comparisons and Satterthwaite’s degrees of freedom method. We used two-tailed tests with a $p < .05$ cutoff for significance.

3. RESULTS

3.1 ASR Errors

3.1.1 Patterns in Error Types

Table 3 provides descriptives on ASR performance measures averaged by student by recording. Immediately apparent is that the vast majority of ASR errors were deletion errors (67%) compared to substitution (17%) and insertion errors (6%; the sum of errors does not add up to 100% because of words correctly recognized). Indeed, when error rate was regressed on error type (three level categorical variable) and number of words in the human transcript (as a covariate), we found the following significant ($ps < .001$) pattern in the errors: Deletion > Substitution > Insertion (Table 3).

3.1.2 Comparing ASR Engines

We regressed each error type on ASR (a three-level categorical effect with Google as the reference group) and reference (human) transcript word count as a covariate. For deletion errors, we found Watson and Rev to be statistically equivalent and higher than Google suggesting the following significant ($ps < .0001$, FDR correction for 3 tests) pattern in the data: [Watson = Rev; $p = .61$] > Google. This pattern was largely flipped for substitution errors: Google > Watson > Rev; $p < .004$. For insertion errors, Google resulted in more insertion errors than Watson, but Rev was intermediate and not significantly different from either. Deletion errors ($p < .001$) were less likely as reference word count increased, but substitution ($p = .582$) and insertion ($p = .137$) errors were not.

Since insertion errors were rare, the tradeoff involved deletion and substitution errors (Figure 3) with Google providing fewest deletions but the most substitutions, the opposite for Rev, and Watson was intermediate. All things equal, the choice of ASR thus depends on obtaining as many transcriptions of speech as possible. Google provided a non-empty transcript for on average 61% of the cases, far exceeding the others (47% for Watson, 41% for Rev), and even among the utterances with nonempty ASR transcriptions, Google had a lower rate of deletions (0.29) than Rev (0.33) and Watson (0.44).

Table 3. Summary statistics of ASR results. M (SD) over utterances

	Google	Rev	Watson
N utterances	1970	1970	1970
N averaged	82	82	82
ASR metrics			
ASR wordcount	2.37 (1.36)	1.71 (1.46)	1.31 (0.95)
Nonempty ASR	0.61 (0.22)	0.41 (0.22)	0.47 (0.22)
Perfect ASR	0.05 (0.06)	0.04 (0.05)	0.02 (0.06)
Insertion rate	0.06 (0.09)	0.08 (0.16)	0.04 (0.08)
Substitution rate	0.21 (0.11)	0.12 (0.08)	0.19 (0.13)
Deletion rate	0.56 (0.18)	0.72 (0.17)	0.72 (0.17)
WER	0.84 (0.15)	0.91 (0.19)	0.95 (0.11)
Downstream NLP metrics			
Topic Distance	0.05 (0.10)	0.05 (0.09)	0.05 (0.07)
BLEU Distance	0.83 (0.11)	0.82 (0.15)	0.94 (0.06)
Semantic Distance	0.56 (0.14)	0.52 (0.15)	0.68 (0.09)
CPS Distance	0.29 (0.14)	0.29 (0.15)	0.33 (0.16)

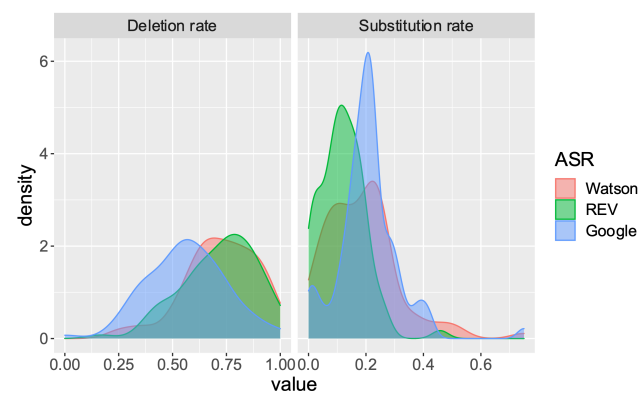


Figure 3. Density plots of deletion and substitution errors by ASR

3.1.3 Sources of Variance

We carried out a multilevel decomposition of variance [35] on each type of ASR error, at three levels: utterance, speaker and recording. Utterances are nested within speakers, and speakers within recordings. We computed the proportion of variance attributable to speaker and recording by decomposing the data into a linear sum of cluster-level averages and within-cluster deviations. The variance between-cluster and within-cluster sums to the total variance, under the assumption that errors at utterance, speaker, and recording are independent. We found that the majority of variance (between 91 and 98%) was at the utterance level for all error types and ASRs, with just 1-3% attributable to individual students and 1-5% to the specific recording (Table 4). This suggests that each ASR system had stable performance across recording contexts and individual differences in vocal parameters.

Table 4. Multilevel variance decomposition. Proportion of variance attributable to each hierarchical level.

ASR	Error type	Utterance	Student	Recording
Google	Insertion rate	0.981	0.009	0.010
Google	Substitution rate	0.980	0.010	0.010
Google	Deletion rate	0.943	0.032	0.025
Rev	Insertion rate	0.980	0.004	0.016
Rev	Substitution rate	0.971	0.009	0.020
Rev	Deletion rate	0.914	0.032	0.053
Watson	Insertion rate	0.976	0.009	0.015
Watson	Substitution rate	0.972	0.011	0.017
Watson	Deletion rate	0.950	0.025	0.025

3.2 Downstream NLP measures

The Spearman correlations between distance metrics for the four downstream tasks are shown in Table 5. The most highly correlated metrics were semantic distance and BLEU distance ($r = .83$), whereas the CPS distance was only moderately correlated (r between .3 and .4) with these measures. Topic distance was not correlated with any other metric, which may be a result of topic words being so rare in the utterance (about 5% of words).

Table 5. Correlations between transcript distance metrics. * $p < 0.001$**

	BLEU Distance	Topic Distance	Semantic Distance
Topic Distance	-0.118		
Semantic Distance	0.830***	-0.040	
CPS Distance	0.321***	0.076	0.402***

Figure 4 shows the distributions of CPS and semantic distances. The peak of the distribution was lower than the baseline (derived by computing the average distances between ASR and human-transcribed utterances after shuffling; see Methods) for all three ASRs and for both CPS and Semantic distances. In fact, 97.5% of semantic distances were less than the shuffled baseline, and 79% for CPS, indicating that a degree of higher-order meaning was generally extracted from the ASR transcripts.

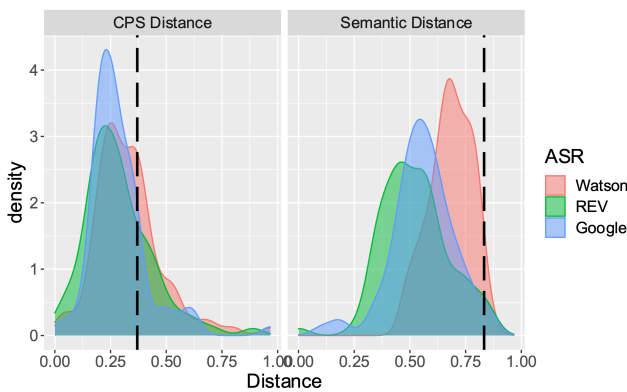


Figure 4. Density plots of CPS Distance and Semantic Distance by ASR. Dashed line shows the random baseline for shuffled utterances.

3.2.1 Comparing ASRs on Downstream NLP

We regressed each distance metric on ASR (a three-level categorical effect with Google as the reference group) and reference (human) transcript word count, with random intercept of student and recording. As indicated in Figure 5, the ASR services did not vary for Topic word distance ($p = .929$), but did for the other measures. Specifically, the pattern of significance ($ps < .001$) for BLEU and semantic distances was: Watson > Google > Rev. For CPS distance it was Watson > Rev, $p = 0.03$; Watson = Google, $p = 0.16$; Rev = Google, $p = 0.46$.

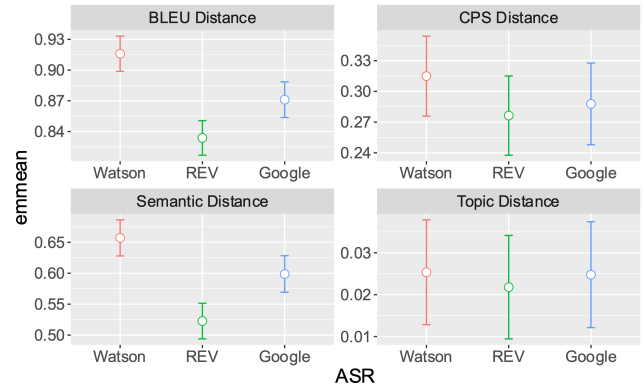


Figure 5. Estimated marginal means and 95% confidence intervals for NLP distance metrics for each of the ASR services

3.2.2 ASR errors on downstream NLP tasks

To test whether specific ASR errors impact downstream NLP metrics, we also fit a linear mixed-effects model to predict each distance metric from the rates of three ASR errors (Table 6). Whereas insertion errors did not significantly predict any of the outcomes, both substitution and deletion errors were negatively ($ps < .001$) associated with BLEU, semantic, and CPS distance measures, more so for the former. Specifically, a one standard deviation increase in each error type was associated with an approximately equivalent increase for BLEU and semantic distances, but only about a half a standard deviation increase for the CPS tasks. Error type was not associated with topic word distance, presumably due to a restriction of range with this measure.

Table 6. Mixed-effects model predicting distance metrics from ASR errors, showing standardized Beta values

Predictors	BLEU Distance		Topic Distance		Semantic Distance		CPS Distance	
	<i>std. Beta</i>	<i>p</i>	<i>std. Beta</i>	<i>p</i>	<i>std. Beta</i>	<i>p</i>	<i>std. Beta</i>	<i>p</i>
(Intercept)	0.07	<0.001	-0.27	0.020	0.05	<0.001	-0.05	0.189
Substitution rate	0.91	<0.001	-0.05	0.231	1.04	<0.001	0.46	<0.001
Deletion rate	1.08	<0.001	-0.05	0.256	1.12	<0.001	0.49	<0.001
Insertion rate	-0.03	0.345	0.01	0.676	0.03	0.446	-0.01	0.870

4. DISCUSSION

We investigated the feasibility of using commercially available ASRs to transcribe student discourse from a collaborative learning activity in a middle school classroom with an eye for downstream NLP tasks aimed to support student learning. In the remainder of this section, we discuss our main findings, applications, limitations, and areas for future work.

4.1 Main Findings

Overall, WER was very high (.84-.95) compared to performance on benchmark datasets, and even compared to WER from prior CL studies using classroom audio, such as in Pugh et al. 2021 who reported a WER of .78 using Watson, but with individual microphones in a more restricted in-class data collection setting compared to the current in-the-wild classroom context. At first blush, these high WERs suggest that it might be futile to expect meaningful ASR in noisy classroom environments without explicitly instrumenting the classroom for this purpose [1] or resorting to miking individual students with customized high-fidelity microphones [90]. However, an in-depth analysis of the pattern of errors suggests that there is hope: specifically, the ASRs had a large proportion of deletion errors and fewer substitution and almost no insertion error meaning that they tended towards high precision but low recall and are thereby feasible for applications that match this profile (as elaborated below).

Comparing the three ASR engines we examined, Google and Rev were biased towards more substitutions and deletions respectively, but also relevant is the proportion of utterances which did not get transcribed at all. Here, Google provided a clear advantage with nonempty results returned for 60% of utterances compared to less than 50% for the other two. Reassuringly, the variance in ASR errors was overwhelmingly from utterance-level differences, with very little attributable to recording or student. In addition, of 1970 utterances, only 477 (24%) returned no transcript from any ASR, raising the possibility of combining outputs from multiple ASRs.

We computed several distance metrics to capture ASR quality as reflected in downstream NLP measures, in each case computing the deviations between ASR-produced and human-transcript versions of each measure. With respect to the four measures, topic word usage was rare and there was very little variability in this measure so unsurprisingly there were no differences for it. Turning to the other measures, Watson was consistently outperformed by Google and Rev, which were equivalent on CPS distance. However, BLEU and semantic distance, which were strongly correlated, were best captured by Rev, despite Google having lower word-level error rates. Thus, Rev had a slight edge over Google for the downstream NLP tasks, but not sufficient to compensate for its higher deletion rate. Finally, as the NLP analyses got more abstract, ASR errors had less of an impact. The effect of substitutions and deletions on CPS distance (a discourse-based construct) was about half that of semantic and BLEU distances, (i.e. word- and semantic-level

measures). Whereas this pattern is intuitively plausible it awaits replication with additional downstream NLP measures.

4.2 Applications

The ability to automatically capture and transcribe student speech during CL activities in the classroom opens the door for numerous applications. Fair and accurate ASR transcripts are the first step for automated interventions that aim to support CL in classrooms. One promising strand of research involves designing teacher-facing applications, such as teacher dashboards, which convey information about student collaborative talk to the teacher. The design space for such technologies is broad and relatively unexplored. While there is potential for abuses such as increased monitoring and evaluation of student talk, responsible innovations can also leverage student transcripts to celebrate students' contributions, build communities within classrooms and foster authentic collaboration motivated by student interest, not desire for positive reinforcement. For example, information gathered from CL discourse could be presented to a teacher offline (i.e., after class), illustrating any number of relevant details about the CL activity (e.g., what students talked about when on-task versus off-task, balance of speaking time, quality of collaboration). To demonstrate, we created an example dashboard visualization of the model-estimated occurrence of three CPS facets in student utterances (Figure 4) using both human- and Google- generated transcripts. As evident in the figure, model estimations are notably impacted by ASR error (i.e., in this group, the model underestimates the use of constructing shared knowledge by students A and C). Although model estimations will be imperfect, they can still provide valuable insights, and the impact of errors can be diminished by aggregating over longer time scales. These after-action reviews could greatly benefit teachers, giving them insight into how they might better support CL in their classroom. This includes designing new activities to better engage students, understanding which student groups may need additional support in future classes and what CPS skills students need help developing.

Similarly, these insights could be conveyed to the teacher online (i.e., during class) via a real-time teacher dashboard. Real-time feedback on CL groups could also enhance a teacher's ability for more effective classroom orchestration by providing them with novel insights into how groups are working together and what kinds of feedback and encouragement will help increase productive collaboration for students. Ultimately, the specifics of these teacher-facing applications, such as what information to present, when to present it (e.g., real-time, offline), how to display it (e.g., graphic representations, transcripts of speech) and at what level of granularity (e.g., individual students, CL groups, whole class) will require co-design, testing, and refinement with teachers.

In addition to teacher-facing applications, ASR systems could be used to create student-facing CL supports in the classroom. These technologies could take many forms, from real-time or after-action

feedback that helps students develop CPS skills to a conversational agent which serves as a socio-collaborative ‘partner’, working together with student groups to enhance learning, equitable participation, and collaboration. Current approaches to support student collaboration (for example by prompting for the use of high-quality discourse called academically productive talk) have been shown to be successful in the context of text chat [85]. Further, after-action reviews to support CPS by providing feedback based on ASR/NLP models has demonstrated potential in the lab [64], but has yet to be tested in classrooms. Whether this is applied to student- or teacher-facing tools, fair and accurate ASR in classrooms has the potential to spotlight students’ verbally-expressed ideas and contributions. This offloads the demand that is normally placed on written work and provides more multimodal dimensions for classroom feedback and support.

Whereas perfect ASR should not be a prerequisite for several applications (as argued in the Introduction), the patterns in ASR errors should be carefully considered in that the ASRs have high precision (relatively low substitution and insertion errors) but low recall (high levels of deletion errors). This suggests that these data are best suited for applications for which transcription of a sampling of utterances is sufficient, for example, assessments of constructs with high-base rates (e.g., CPS skills) rather than those focused on rare events. This high precision could be helpful in avoiding unwarranted interventions triggered by CL supports, as there should be a low rate of false alarms of discourse features detected based on the ASR results. Nevertheless, our findings suggest that a real-time conversational partner will likely be off the table until ASR deletion errors can be reduced. Nevertheless, robustness of NLP models to ASR errors can be improved by data augmentation approaches where models are trained on ASR hypotheses as well as human transcripts [58].

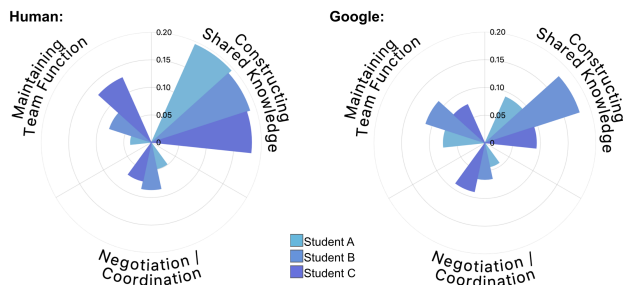


Figure 5. Predicted probabilities of CPS skills over a sample 5-minute recording (using the student group shown in Table 2) based on human (left) and Google ASR (right) transcripts

4.3 Limitations and Future Work

There were several limitations to this study. First, although we investigated an automated approach to transcribe student utterances, we did not incorporate automatic utterance segmentation in our pipeline. Rather, utterances were manually identified and segmented by a human observer before being processed by the ASR systems, which is consistent with prior work on comparing ASRs [5, 18]. This was done because the present focus was on speech-to-text transcription and not utterance-segmentation, so we opted for a gold-standard baseline for the latter to compare the various ASRs for the former. Further, utterance-segmentation is technically not needed as a separate step in an automated pipeline in that the entire five-minute audio segment could be submitted to the ASR engines for combined utterance

segmentation and speech transcription, albeit less accurately than human segmentation. Indeed, longer context than single utterances are beneficial in modeling CL [64].

Another limitation is that we only tested out-of-the-box cloud-based ASR systems. One problem with this approach is that reliance on cloud-based services may be unrealistic in the near-term. In the US, nearly 28 million students did not have sufficient internet bandwidth for multimedia learning [97]. Similarly, we did not attempt to improve the performance of these out-of-the-box systems (e.g., by fine-tuning a custom ASR model on our data or providing a task-specific vocabulary) because the present goal was to compare these systems “as-is” since many researchers might not have the technical expertise needed to train customized models or fine-tune existing models. However, recent advances in deep-learning-based ASR mean pretrained models are widely available and a relatively small amount of data is needed for fine tuning [2, 74], which may provide better performance in this domain than standard cloud-based systems. To this point, we are currently developing a customized, locally hosted ASR system to improve upon the present results and address the limitations above.

One additional limitation is the lack of diversity in our sample. Whereas student-level demographic data was unavailable, district-level information suggests that 92% of the students were either White (62%) or Hispanic (30%). Racial disparities in ASR performance [44], as well as challenges with non-native English speakers [93] are well documented and may have disproportionately adverse effects on underrepresented groups when ASR is used for downstream applications. Thus, the lack of variability at the student level might be partly because our sample was non-representative. To create more fair ASR transcripts, non-native English speakers and students from non-dominant cultures should be oversampled to create representation, and thus accuracy, equal to students from dominant cultures. We also chose to include data from a single (although multi-lesson and multi-day) curriculum unit as implemented by one teacher with a small number of students. In sum, these factors reduce the generalizability of our findings to groups historically underrepresented in STEM. Our future work will aim to address these limitations by collecting classroom speech from racially and socioeconomically diverse populations, and examining ASR performance across different groups to identify sources of bias or nonequivalence.

4.4 Conclusion

Automated speech recognition in conjunction with natural language processing has the potential to unlock collaborative learning supports in the classroom. We recorded authentic small-group interactions in middle school STEM classrooms using inexpensive, commercially-available equipment, and analyzed the transcripts provided by several cloud providers. We show how different types of transcription errors influence downstream linguistic models, and find that the impact of ASR errors is smaller for the predictive accuracy of a CL model than for upstream measures capturing more literal aspects of speech content. Our results demonstrate the challenges of automating speech recognition in the classroom, but suggest the potential of using imperfect ASR to gain insights into collaborative discourse.

5. ACKNOWLEDGMENTS

This research was supported by the NSF National AI Institute for Student-AI Teaming (iSAT) (DRL 2019805). The opinions expressed are those of the authors and do not represent views of the funding agencies. We also thank Jon Cai (LDA topic models) and Robert Moulder (variance decomposition).

6. REFERENCES

- [1] Ahuja, K., Kim, D., Xhakaj, F., Varga, V., Xie, A., Zhang, S., Townsend, J.E., Harrison, C., Ogan, A. and Agarwal, Y. 2019. EduSense: Practical Classroom Sensing at Scale. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*. 3, 3 (2019), 1–26. DOI:https://doi.org/10.1145/3351229.
- [2] Baevski, A., Zhou, H., Mohamed, A. and Auli, M. 2020. wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations. *arXiv*. (2020).
- [3] Bai, Y., Tejedor-García, C., Hubers, F., Cucchiari, C. and Strik, H. 2021. An ASR-Based Tutor for Learning to Read: How to Optimize Feedback to First Graders. (2021), 58–69.
- [4] Bidy, Q., Chakarov, A.G., Bush, J., Elliott, C.H., Jacobs, J., Recker, M., Sumner, T. and Penuel, W. 2021. Designing a Middle School Science Curriculum that Integrates Computational Thinking and Sensor Technology. *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*. (Dec. 2021), 818–824. DOI:https://doi.org/10.1145/3287324.3287476.
- [5] Blanchard, N., Brady, M., Olney, A.M., Glaus, M., Sun, X., Nystrand, M., Samei, B., Kelly, S. and D’Mello, S. 2015. A Study of Automatic Speech Recognition in Noisy Classroom Environments for Automated Dialog Analysis. *Proceedings of the 8th International Conference on Educational Data Mining 283* (2015), 23–33.
- [6] Blanchard, N., D’Mello, S., Olney, A.M. and Nystrand, M. 2015. Automatic Classification of Question & Answer Discourse Segments from Teacher’s Speech in Classrooms. *Proceedings of the 8th International Conference on Educational Data Mining*. (2015), 282–288.
- [7] Blanchard, N., Donnelly, P., Olney, A.M., Samei, B., Ward, B., Sun, X., Kelly, S., Nystrand, M. and D’Mello, S.K. 2016. Semi-Automatic Detection of Teacher Questions from Human-Transcripts of Audio in Live Classrooms. (2016).
- [8] Bransford, J.D., Brown, A.L. and Cocking, R.R. 2000. How People Learn: Brain, Mind, Experience, and School.
- [9] Caballero, D., Araya, R., Kronholm, H., Viiri, J., Mansikkaniemi, A., Lehesvuori, S., Virtanen, T., & Kurimo, M. (2017). Data Driven Approaches in Digital Education, 12th European Conference on Technology Enhanced Learning, EC-TEL 2017, Tallinn, Estonia, September 12–15, 2017, Proceedings. *Lecture Notes in Computer Science*, 541–544. https://doi.org/10.1007/978-3-319-66610-5_58
- [10] Cazden, C.B. 1988. *Classroom discourse: The language of teaching and learning*. ERIC.
- [11] Chakarov, A.G., Bidy, Q., Elliott, C.H. and Recker, M. 2021. The Data Sensor Hub (DaSH): A Physical Computing System to Support Middle School Inquiry Science Instruction. *Sensors (Basel, Switzerland)*. 21, 18 (2021), 6243. DOI:https://doi.org/10.3390/s21186243.
- [12] Chang, X., Zhang, W., Qian, Y., Roux, J.L. and Watanabe, S. 2020. End-To-End Multi-Speaker Speech Recognition With Transformer. *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 00, (2020), 6134–6138. DOI:https://doi.org/10.1109/icassp40776.2020.9054029.
- [13] Cole, R., Buchenroth-Martin, C., Weston, T., Devine, L., Myatt, J., Holding, B., Pradhan, S., McKeown, M., Messier, S., Borum, J. and Ward, W. 2018. One-on-one and small group conversations with an intelligent virtual science tutor. *Computer Speech & Language*. 50, (2018), 157–174. DOI:https://doi.org/10.1016/j.csl.2018.01.002.
- [14] Cook, C., Olney, A.M., Kelly, S. and D’Mello, S.K. 2018. An Open Vocabulary Approach for Estimating Teacher Use of Authentic Questions in Classroom Discourse. *Proceedings of the 11th International Conference on Educational Data Mining*. (Jun. 2018), 116–126.
- [15] Deng, A. 2021. *Fostering Literacy with Speech Recognition: A Pilot Study*.
- [16] Devlin, J., Chang, M.-W., Lee, K. and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. (2019), 4171–4186. DOI:https://doi.org/10.18653/v1/n19-1423.
- [17] Dillenbourg, P. (1999). What do you mean by collaborative learning? Chapter 1. In Dillenbourg (Ed.), *Collaborative-learning: Cognitive and Computational Approaches*. (Vol. 1, pp. 1–19). Oxford: Elsevier.
- [18] D’Mello, S.K., Olney, A.M., Blanchard, N., Samei, B., Sun, X., Ward, B. and Kelly, S. 2015. Multimodal Capture of Teacher-Student Interactions for Automated Dialogic Analysis in Live Classrooms. *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*. (2015), 557–566. DOI:https://doi.org/10.1145/2818346.2830602.
- [19] Feng, S., Kudina, O., Halpern, B.M. and Scharenborg, O. 2021. Quantifying Bias in Automatic Speech Recognition. *arXiv*. (2021).
- [20] Filippidou, F. and Moussiades, L. 2020. A Benchmarking of IBM, Google and Wit Automatic Speech Recognition Systems. *Artificial Intelligence Applications and Innovations*. 583, (2020), 73–82. DOI:https://doi.org/10.1007/978-3-030-49161-1_7.
- [21] Flor, M., Yoon, S.-Y., Hao, J., Liu, L. and Davier, A. von 2016. Automated classification of collaborative problem solving interactions in simulated science tasks. *Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications*. (2016), 31–41. DOI:https://doi.org/10.18653/v1/w16-0504.
- [22] Foltz, P.W., Laham, D. and Derr, M. 2003. Automated Speech Recognition for Modeling Team Performance. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*. 47, 4 (2003), 673–677. DOI:https://doi.org/10.1177/154193120304700402.
- [23] Gamper, H., Emmanouilidou, D., Braun, S. and Tashev, I.J. 2020. Predicting Word Error Rate for Reverberant Speech. *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 00, (2020), 491–495. DOI:https://doi.org/10.1109/icassp40776.2020.9053025.
- [24] German, S. 2019. Using the Anchoring Phenomenon Routine to introduce a science unit. *Science Scope*. 42, 5 (2019), 32–35.
- [25] Gerosa, M., Giuliani, D., Narayanan, S. and Potamianos, A. 2009. A review of ASR technologies for children’s speech.

- Proceedings of the 2nd Workshop on Child, Computer and Interaction - WOCCI '09.* (2009), 7.
DOI:<https://doi.org/10.1145/1640377.1640384>.
- [26] Google Cloud Speech-to-Text: <https://cloud.google.com/speech-to-text/>. Accessed: 2022-03-04.
- [27] Graesser, A. C., Greiff, S., Stadler, M., & Shubeck, K. T. (2019). Collaboration in the 21st Century: The Theory, Assessment, and Teaching of Collaborative Problem Solving. *Computers in Human Behavior*, 104, 106134. <https://doi.org/10.1016/j.chb.2019.09.010>
- [28] Graesser, A.C., Person, N.K. and Magliano, J.P. 1995. Collaborative dialogue patterns in naturalistic one-to-one tutoring. *Applied Cognitive Psychology*. 9, 6 (1995). DOI:<https://doi.org/10.1002/acp.2350090604>.
- [29] Griffin, P., Care, E. and McGaw, B. The changing role of education and schools. In P. Griffin, B. McGaw, & E. Care (Eds.), *Assessment and Teaching of 21st Century Skills* (pp. 1-16). Dordrecht, Germany: Springer
http://dx.doi.org/10.1007/978-94-007-2324-5_2
- [30] Hao, J., Chen, L., Flor, M., Liu, L. and Davier, A.A. von 2017. CPS-Rater: Automated Sequential Annotation for Conversations in Collaborative Problem-Solving Activities. *ETS Research Report Series*. 2017, 1 (2017), 1–9. DOI:<https://doi.org/10.1002/ets2.12184>.
- [31] Hardy, L., Dixon, C. and Hsi, S. 2019. From Data Collectors to Data Producers: Shifting Students' Relationship to Data. *Journal of the Learning Sciences*. 29, 1 (2019), 1–23. DOI:<https://doi.org/10.1080/10508406.2019.1678164>.
- [32] Henri, F. 1992. Computer Conferencing and Content Analysis. *Collaborative Learning Through Computer Conferencing*.
- [33] Hmelo-Silver, C. E., & Barrows, H. S. (2008). Facilitating collaborative knowledge building. *Cognition and Instruction*, 26(1), 48–94. <https://doi.org/10.1080/07370000701798495>
- [34] Howard, C.S., Munro, K.J. and Plack, C.J. 2010. Listening effort at signal-to-noise ratios that are typical of the school classroom. *International Journal of Audiology*. 49, 12 (2010), 928–932. DOI:<https://doi.org/10.3109/14992027.2010.520036>.
- [35] Husson, F., Josse, J., Narasimhan, B. and Robin, G. 2019. Imputation of Mixed Data With Multilevel Singular Value Decomposition. *Journal of Computational and Graphical Statistics*. 28, 3 (2019), 552–566. DOI:<https://doi.org/10.1080/10618600.2019.1585261>.
- [36] IBM Watson: <https://www.ibm.com/watson/services/speech-to-text/>. Accessed: 2022-03-04.
- [37] Jensen, E., Dale, M., Donnelly, P.J., Stone, C., Kelly, S., Godley, A. and D'Mello, S.K. 2020. Toward Automated Feedback on Teacher Discourse to Enhance Teacher Learning. *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. (2020), 1–13. DOI:<https://doi.org/10.1145/3313831.3376418>.
- [38] Jensen, E., Pugh, S.L. and D'Mello, S.K. 2021. A Deep Transfer Learning Approach to Modeling Teacher Discourse in the Classroom. *LAK21: 11th International Learning Analytics and Knowledge Conference*. (2021), 302–312. DOI:<https://doi.org/10.1145/3448139.3448168>.
- [39] Jeong, H., & Hmelo-Silver, C. E. (2016). Seven Affordances of Computer-Supported Collaborative Learning: How to Support Collaborative Learning? How Can Technologies Help? *Educational Psychologist*, 51(2), 247–265. <https://doi.org/10.1080/00461520.2016.1158654>
- [40] Jiang, D., Chen, Y. and Garg, A. 2018. A hybrid method for overlapping speech detection in classroom environment. *Computer Applications in Engineering Education*. 26, 1 (2018), 171–180. DOI:<https://doi.org/10.1002/cae.21855>.
- [41] Johnson, R., Severance, S., Penuel, W.R. and Leary, H. 2016. Teachers, tasks, and tensions: lessons from a research–practice partnership. *Journal of Mathematics Teacher Education*. 19, 2–3 (2016), 169–185. DOI:<https://doi.org/10.1007/s10857-015-9338-3>.
- [42] Kelly, S., Olney, A.M., Donnelly, P., Nystrand, M. and D'Mello, S.K. 2018. Automatically Measuring Question Authenticity in Real-World Classrooms. *Educational Researcher*. 47, 7 (2018), 451–464. DOI:<https://doi.org/10.3102/0013189x18785613>.
- [43] Kim, S., Arora, A., Le, D., Yeh, C.-F., Fuegen, C., Kalinli, O. and Seltzer, M.L. 2021. Semantic Distance: A New Metric for ASR Performance Analysis Towards Spoken Language Understanding. *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH* (2021), 1977–1981.
- [44] Koenecke, A., Nam, A., Lake, E., Nudell, J., Quartey, M., Mengesha, Z., Toups, C., Rickford, J.R., Jurafsky, D. and Goel, S. 2020. Racial disparities in automated speech recognition. *Proceedings of the National Academy of Sciences of the United States of America*. 117, 14 (2020), 7684–7689. DOI:<https://doi.org/10.1073/pnas.1915768117>.
- [45] Koller, M. (2016). robustlmm : An R Package for Robust Estimation of Linear Mixed-Effects Models. *Journal of Statistical Software*, 75(6), 1–24. <https://doi.org/10.18637/jss.v075.i06>
- [46] Lee, S., Potamianos, A. and Narayanan, S. 1999. Acoustics of children's speech: Developmental changes of temporal and spectral parameters. *The Journal of the Acoustical Society of America*. 105, 3 (1999), 1455–1468. DOI:<https://doi.org/10.1121/1.426686>.
- [47] Li, H., Ding, W. and Liu, Z. 2020. Identifying At-Risk K-12 Students in Multimodal Online Environments: A Machine Learning Approach. *arXiv*. (2020).
- [48] Lileikyte, R., Irvin, D. and Hansen, J.H.L. 2020. Assessing Child Communication Engagement via Speech Recognition in Naturalistic Active Learning Spaces. *The Speaker and Language Recognition Workshop (Odyssey 2020)*. (2020), 396–401. DOI:<https://doi.org/10.21437/odyssey.2020-56>.
- [49] Ling, H., Han, P., Qiu, J., Peng, L., Liu, D. and Luo, K. 2021. A Method of Speech Separation between Teachers and Students in Smart Classrooms Based on Speaker Diarization. *2021 13th International Conference on Education Technology and Computers*. (2021), 53–61. DOI:<https://doi.org/10.1145/3498765.3498774>.
- [50] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L. and Stoyanov, V. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv*. (2019).

- [51] Loukina, A., Madnani, N., Klebanov, B.B., Misra, A., Angelov, G. and Todic, O. 2018. Evaluating on-device ASR on Field Recordings from an Interactive Reading Companion. *2018 IEEE Spoken Language Technology Workshop (SLT)*. 00, (2018), 964–970. DOI:<https://doi.org/10.1109/slt.2018.8639603>.
- [52] Lugini, L., Olshefski, C., Singh, R., Litman, D. and Godley, A. 2020. Discussion Tracker: Supporting Teacher Learning about Students' Collaborative Argumentation in High School Classrooms. *Proceedings of the 28th International Conference on Computational Linguistics: System Demonstrations*. (2020), 53–58. DOI:<https://doi.org/10.18653/v1/2020.coling-demos.10>.
- [53] Ma, Y., Wiggins, J.B., Celepkolu, M., Boyer, K.E., Lynch, C. and Wiebe, E. 2021. The Challenge of Noisy Classrooms: Speaker Detection During Elementary Students' Collaborative Dialogue. (2021), 268–281.
- [54] MacNeilley, L.H., Nystrand, M., Gamoran, A., Kachur, R. and Prendergast, C. 1998. Opening Dialogue: Understanding the Dynamics of Language and Learning in the English Classroom. *Language*. 74, 2 (1998), 444. DOI:<https://doi.org/10.2307/417942>.
- [55] Manning, C., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S. and McClosky, D. 2020. The Stanford CoreNLP Natural Language Processing Toolkit. *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. 55–60. DOI:<https://doi.org/10.3115/v1/p14-5010>.
- [56] Mutlu, B., Tscheligi, M., Weiss, A., Young, J.E., Kennedy, J., Lemaignan, S., Montassier, C., Lavalade, P., Irfan, B., Papadopoulos, F., Senft, E. and Belpaeme, T. 2017. Child Speech Recognition in Human-Robot Interaction. *Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*. (2017), 82–90. DOI:<https://doi.org/10.1145/2909824.3020229>.
- [57] National Teacher and Principal Survey: 2022. https://nces.ed.gov/surveys/ntps/tables/ntps1718_ftable06_t1s.asp.
- [58] Nechaev, Y., Ruan, W. and Kiss, I. 2021. Towards NLU Model Robustness to ASR Errors at Scale. (2021).
- [59] Papineni, K., Roukos, S., Ward, T. and Zhu, W.-J. 2002. BLEU: a method for automatic evaluation of machine translation. *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics - ACL '02*. (2002), 311–318. DOI:<https://doi.org/10.3115/1073083.1073135>.
- [60] Pascual, R.M. 2020. Effectiveness of an Automated Reading Tutor Design for Filipino Speaking Children. *2020 IEEE 8th R10 Humanitarian Technology Conference (R10-HTC)*. 00, (2020), 1–5. DOI:<https://doi.org/10.1109/r10-htc49770.2020.9357059>.
- [61] Potamianos, A. and Narayanan, S. 2003. Robust recognition of children's speech. *IEEE Transactions on Speech and Audio Processing*. 11, 6 (2003), 603–616. DOI:<https://doi.org/10.1109/tsa.2003.818026>.
- [62] Praharaj, S., Scheffel, M., Drachler, H. and Specht, M. 2019. Literature Review on Co-Located Collaboration Modeling Using Multimodal Learning Analytics Can We Go the Whole Nine Yards? *IEEE Transactions on Learning Technologies*. 14, 3 (2019), 367–385. DOI:<https://doi.org/10.1109/tlt.2021.3097766>.
- [63] Pugh, S.L., Rao, A.R., Stewart, A.E.B. and D'Mello, S.K. 2022. Do Speech-Based Collaboration Analytics Generalize Across Task Contexts? *Learning and Knowledge* 22. (Apr. 2022), 208–218.
- [64] Pugh, S.L., Subburaj, S.K., Rao, A.R., Stewart, A.E.B., Andrews-Todd, J. and D'Mello, S.K. 2021. Say What? Automatic Modeling of Collaborative Problem Solving Skills from Student Speech in the Wild. *Proceedings of The 14th International Conference on Educational Data Mining (EDM 2021)*. (Mar. 2021), 55–67.
- [65] Qian, Y., Weng, C., Chang, X., Wang, S. and Yu, D. 2018. Past review, current progress, and challenges ahead on the cocktail party problem. *Frontiers of Information Technology & Electronic Engineering*. 19, 1 (2018), 40–63. DOI:<https://doi.org/10.1631/fit.1700814>.
- [66] Reeder, K., Shapiro, J., Wakefield, J. and D'Silva, R. 2015. Speech Recognition Software Contributes to Reading Development for Young Learners of English. *International Journal of Computer-Assisted Language Learning and Teaching (IJCALLT)*. 5, 3 (2015), 60–74. DOI:<https://doi.org/10.4018/ijcallt.2015070104>.
- [67] Reimers, N. and Gurevych, I. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. *arXiv*. (2019).
- [68] Rev Speech-to-Text API: <https://www.rev.ai/>. Accessed: 2022-03-04.
- [69] Rev vs Google ASR: <https://www.rev.com/blog/google-speech-recognition-api-vs-rev-ai-api>. Accessed: 2022-03-11.
- [70] Rodrigues, A., Santos, R., Abreu, J., Beça, P., Almeida, P. and Fernandes, S. 2019. Analyzing the performance of ASR systems. *Proceedings of the XX International Conference on Human Computer Interaction*. (2019), 1–8. DOI:<https://doi.org/10.1145/3335595.3335635>.
- [71] Roschelle, J., Dimitriadis, Y., and Hoppe, U. (2013). Classroom orchestration: Synthesis. *Computers & Education*, 69, 523–526. <https://doi.org/10.1016/j.compedu.2013.04.010>
- [72] Roschelle, J. and Teasley, S.D. 1995. Computer Supported Collaborative Learning. *Computer Supported Collaborative Learning*. 69–97.
- [73] Rosé, C., Wang, Y.-C., Cui, Y., Arguello, J., Stegmann, K., Weinberger, A. and Fischer, F. 2008. Analyzing collaborative learning processes automatically: Exploiting the advances of computational linguistics in computer-supported collaborative learning. *International Journal of Computer-Supported Collaborative Learning*. 3, 3 (2008), 237–271. DOI:<https://doi.org/10.1007/s11412-007-9034-0>.
- [74] Shivakumar, P.G. and Narayanan, S. 2022. End-to-end neural systems for automatic children speech recognition: An empirical study. *Computer Speech & Language*. 72, (2022), 101289. DOI:<https://doi.org/10.1016/j.csl.2021.101289>.
- [75] Smith, L. and Macgregor, J.T. 1992. What is Collaborative Learning? *Assessment*. 117, 5 (1992), 1–11.
- [76] Song, Y., Lei, S., Hao, T., Lan, Z. and Ding, Y. 2021. Automatic Classification of Semantic Content of Classroom Dialogue. *Journal of Educational Computing Research*. 59, 3 (2021), 496–521. DOI:<https://doi.org/10.1177/0735633120968554>.

- [77] St Vrain Demographics: https://edx.cde.state.co.us/SchoolView/DataCenter/reports.jsp?Dis=0470&_afzLoop=3057061758625614&_afzWindowMode=0&tab=pro&_adf.ctrl-state=nxjlz2w8s_4. Accessed: 2022-03-09.
- [78] Stewart, A.E.B., Keirn, Z. and D’Mello, S.K. 2021. Multimodal modeling of collaborative problem-solving facets in triads. *User Modeling and User-Adapted Interaction*. 31, 4 (2021), 713–751. DOI:<https://doi.org/10.1007/s11257-021-09290-y>.
- [79] Stewart, A.E.B., Vrzakova, H., Sun, C., Yonehiro, J., Stone, C.A., Duran, N.D., Shute, V. and D’Mello, S.K. 2019. I Say, You Say, We Say. *Proceedings of the ACM on Human-Computer Interaction*. 3, CSCW (Sep. 2019), 1–19. DOI:<https://doi.org/10.1145/3359296>.
- [80] Sun, C., Shute, V.J., Stewart, A., Yonehiro, J., Duran, N. and D’Mello, S. 2020. Towards a generalized competency model of collaborative problem solving. *Computers & Education*. 143, (2020), 103672. DOI:<https://doi.org/10.1016/j.compedu.2019.103672>.
- [81] Suresh, A., Jacobs, J., Lai, V., Tan, C., Ward, W., Martin, J.H. and Sumner, T. 2021. Using Transformers to Provide Teachers with Personalized Feedback on their Classroom Discourse: The TalkMoves Application. *arXiv*. (2021).
- [82] Suresh, A., Sumner, T., Jacobs, J., Foland, B. and Ward, W. 2019. Automating Analysis and Feedback to Improve Mathematics Teachers’ Classroom Discourse. *Proceedings of the AAAI Conference on Artificial Intelligence*. 33, (2019), 9721–9728. DOI:<https://doi.org/10.1609/aaai.v33i01.33019721>.
- [83] Szymański, P., Żelasko, P., Morzy, M., Szymczak, A., Żyła-Hoppe, M., Banaszczak, J., Augustyniak, L., Mizgajski, J. and Carmiel, Y. 2020. WER we are and WER we think we are. *arXiv*. (2020).
- [84] Tao, Y., Mitsven, S.G., Perry, L.K., Messinger, D.S. and Shyu, M.-L. 2019. Audio-Based Group Detection for Classroom Dynamics Analysis. *2019 International Conference on Data Mining Workshops (ICDMW)*. 00, (2019), 855–862. DOI:<https://doi.org/10.1109/icdmw.2019.00125>.
- [85] Tegos, S., Demetriadis, S. and Karakostas, A. 2015. Promoting academically productive talk with conversational agent interventions in collaborative learning settings. *Computers & Education*. 87, (2015), 309–325. DOI:<https://doi.org/10.1016/j.compedu.2015.07.014>.
- [86] Terenzini, P.T., Cabrera, A.F., Colbeck, C.L., Parente, J.M. and Bjorklund, S.A. 2001. Collaborative Learning vs. Lecture/Discussion: Students’ Reported Learning Gains*. *Journal of Engineering Education*. 90, 1 (2001), 123–130. DOI:<https://doi.org/10.1002/j.2168-9830.2001.tb00579.x>.
- [87] Tissenbaum, M. and Slotta, J.D. 2019. Developing a smart classroom infrastructure to support real-time student collaboration and inquiry: a 4-year design study. *Instructional Science*. 47, 4 (2019), 423–462. DOI:<https://doi.org/10.1007/s11251-019-09486-1>.
- [88] Tissenbaum, M. and Slotta, J.D. 2015. Seamless Learning in the Age of Mobile Connectivity. *Seamless Learning in the Age of Mobile Connectivity*. 223–257.
- [89] Vygotsky, L.S. 1978. *Mind and society: The Development of Higher Mental Processes*. Harvard University Press.
- [90] Wang, Z., Miller, K. and Cortina, K. 2013. Using the LENA in Teacher Training: Promoting Student Involvement through automated feedback. *Unterrichtswissenschaft*. 4, (2013), 290–305.
- [91] Wang, Z., Pan, X., Miller, K.F. and Cortina, K.S. 2014. Automatic classification of activities in classroom discourse. *Computers & Education*. 78, (2014), 115–123. DOI:<https://doi.org/10.1016/j.compedu.2014.05.010>.
- [92] Ward, W., Cole, R., Bolaños, D., Buchenroth-Martin, C., Svirsky, E., Vuuren, S.V., Weston, T., Zheng, J. and Becker, L. 2011. My science tutor: A conversational multimedia virtual tutor for elementary school science. *ACM Transactions on Speech and Language Processing (TSLP)*. 7, 4 (2011), 18. DOI:<https://doi.org/10.1145/1998384.1998392>.
- [93] Wu, Y., Rough, D., Bleakley, A., Edwards, J., Cooney, O., Doyle, P.R., Clark, L. and Cowan, B.R. 2020. See What I’m Saying? Comparing Intelligent Personal Assistant Use for Native and Non-Native Language Speakers. *22nd International Conference on Human-Computer Interaction with Mobile Devices and Services*. (2020), 1–9. DOI:<https://doi.org/10.1145/3379503.3403563>.
- [94] Xu, S., Ding, W. and Liu, Z. 2020. Automatic Dialogic Instruction Detection for K-12 Online One-on-One Classes. *Artificial Intelligence in Education*. 12164, (2020), 340–345. DOI:https://doi.org/10.1007/978-3-030-52240-7_62.
- [95] Yasin, I., Liu, F., Drga, V., Demosthenous, A. and Meddis, R. 2018. Effect of auditory efferent time-constant duration on speech recognition in noise. *The Journal of the Acoustical Society of America*. 143, 2 (2018), EL112–EL115. DOI:<https://doi.org/10.1121/1.5023502>.
- [96] Zyllich, B. and Whitehill, J. 2020. Noise-Robust Key-Phrase Detectors for Automated Classroom Feedback. *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 00, (2020), 9215–9219. DOI:<https://doi.org/10.1109/icassp40776.2020.9053173>.
- [97] *How people learn II: Learners, contexts, and cultures*. 2021. *REPORT ON SCHOOL CONNECTIVITY FUNDING YEAR 2021*. Connect K-12.

Does Practice Make Perfect? Analyzing the Relationship Between Higher Mastery and Forgetting in an Adaptive Learning System

Jeffrey Matayoshi
McGraw Hill ALEKS
jeffrey.matayoshi
@mheducation.com

Eric Cosyn
McGraw Hill ALEKS
eric.cosyn
@mheducation.com

Hasan Uzun
McGraw Hill ALEKS
hasan.uzun
@mheducation.com

ABSTRACT

As outlined by Benjamin Bloom, students working within a mastery learning framework must demonstrate mastery of the core prerequisite material before learning any subsequent material. Since many learning systems in use today adhere to these principles, an important component of such systems is the set of rules or algorithms that determine when a student has demonstrated mastery. A relevant issue when discussing mastery learning is its durability—in particular, we are interested in the relationship between different mastery learning thresholds and the forgetting of the learned material. As such, in this study we investigate this question using a large data set from the ALEKS adaptive learning system. Applying a quasi-experimental design, we find evidence that, while a higher mastery threshold is initially associated with a higher rate of knowledge retention, after several weeks this difference has largely disappeared.

Keywords

Mastery learning, forgetting, adaptive learning

1. INTRODUCTION

Many adaptive learning and intelligent tutoring systems in use today employ the principles of *mastery learning*. As outlined by Benjamin Bloom [9], in such a framework students must demonstrate mastery of the core prerequisite material before working on any subsequent material. Thus, an important component of any system implementing mastery learning is the set of rules or algorithms used to determine when a student has mastered a skill or problem type. Over the years, important families of models have been developed for this purpose, with perhaps the most noteworthy being Bayesian knowledge tracing (BKT) and its derivatives [6, 17, 43, 68], and the factors analysis family of models, with examples of the latter including Learning Factors Analysis (LFA) [13] and Performance Factors Analysis (PFA) [45]. Additionally, other simpler rules and heuristics, such as re-

quiring students to correctly answer a certain number of questions in a row [32], are also utilized.¹

As there is a balance between ensuring students have sufficiently mastered a problem type, while not subjecting them to more practice than necessary—variously referred to as “over practice” [14] or “overlearning” [51]—previous works have looked in detail at mastery learning thresholds and how to optimize them for various factors such as student learning efficiency [7, 14] and classification performance [22, 32]. Additionally, it has been argued that the choice of data and the threshold used are more important than the specific type of model being applied [46].

A related subject is that of knowledge retention and forgetting. In particular, the Ebbinghaus forgetting curve [4, 21] models the decay of knowledge over time, with numerous studies having looked at the conditions affecting these curves in settings as varied as laboratory experiments [26, 40, 42, 56], classrooms [2, 8, 25], and adaptive learning and intelligent tutoring systems [37, 38, 62, 65, 66]. Other works have shown that learning systems benefit greatly by accounting for forgetting [16, 35, 47, 63] and having personalized interventions and review schedules [34, 44, 55, 58, 67].

In this work, we are interested in the relationship between different mastery thresholds and the retention of knowledge. Additionally, we compare and contrast the frequencies at which problem types are successfully learned under these mastery thresholds. To perform these analyses, we take advantage of a “natural” experiment that occurs within the ALEKS adaptive learning system where, depending upon the outcome of an assessment given at the beginning of a course, problem types are assigned to two different mastery thresholds. By comparing the outcomes from these different thresholds, we hope to understand more about the relationship between higher mastery, extra practice, and forgetting.

2. BACKGROUND

In this section we give a brief background of the ALEKS system. Within the system, a *topic* is a problem type that covers a discrete unit of an academic course. Each topic contains many examples called *instances*, with these examples being chosen so that they cover the same content and are

¹Interestingly, recent work has shown that some of these simpler models—including the one we consider in this study—can be viewed as special cases of BKT [19].

J. Matayoshi, E. Cosyn, and H. Uzun. Does practice make perfect? Analyzing the relationship between higher mastery and forgetting in an adaptive learning system. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 316–324, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853075>

equal in difficulty. The topics in an ALEKS course contain many *prerequisite* relationships. That is, topic x is a prerequisite for topic y if x contains certain core concepts and material that must be learned before it's possible to learn the material in y .

At the start of an ALEKS course, the student's incoming knowledge is measured by an adaptive *initial assessment*. After each question of this assessment, a probability estimate is computed for each topic in the course, with this probability measuring how likely it is that the student knows the topic. At the end of the assessment, based on both these probability estimates and the prerequisite relationships between the topics, the ALEKS system partitions the topics in the course into the following three categories.²

- Topics that are most likely known
- Topics that are most likely unknown
- All remaining topics (uncertain)

Next, in the ALEKS learning mode a student is presented a topic the system believes they are ready to learn, and the student can access additional topics they are ready to learn from a graphical list. In all cases, the topics being learned are from the uncertain and unknown categories. To determine mastery, a *high mastery* threshold is used for the unknown topics, while a *low mastery* threshold is used for the uncertain topics (we give precise definitions of these thresholds shortly). As the system is not sure if the uncertain topics are actually known by the student, relaxing the threshold allows the student to more quickly demonstrate mastery.

When learning a topic, a student can take three possible actions: submitting a correct answer, submitting a wrong answer, or accessing an explanation page with a worked solution to the current instance. We define the *learning sequence* to be the sequence of actions taken by the student while working on a particular topic. A student begins their learning sequence with a score of 0, whereupon they are presented an example instance with a worked explanation. Following this, the student receives another instance for actual practice. Each time the student receives a new instance, they can try to answer it, or they can access the explanation page. Note that a student is always given a new instance after a correct answer, viewing an explanation, or submitting two consecutive wrong answers. Depending on the student's action, the score is updated based on the following rules.

- (1) A single correct answer increases the score by 1; however, if the correct answer immediately follows a previous correct answer, the score increases by 2 instead.
- (2) An incorrect answer decreases the score by 1 (unless the score is already at 0).
- (3) Viewing an explanation does not change the score. However, it does affect rule (1)—for example, if a student answers correctly immediately after viewing an explanation, the score increases by only 1 point, rather than 2, regardless of the student's previous responses.

²While beyond the scope of this study, the validity of both the probability estimates and the topic categorizations have been evaluated in works such as [18, 39].

For an unknown topic, the student must achieve a score of 5 before the topic is considered mastered—this is the aforementioned high mastery threshold. Alternatively, topics that are classified as uncertain only require a score of 3 to achieve mastery—this is the low mastery threshold. Finally, if a student gives five consecutive incorrect answers, this is considered to be a failed learning attempt—in such a case, the student is gently prompted to try another topic.

3. EXPERIMENTAL SETUP

Our study uses data from 13 different ALEKS mathematics products, ranging from elementary school mathematics to college-level algebra. From these products, we gathered data for a total of 2,235,061 students over a three-year period starting in January 2017. While we don't have access to detailed demographic information for our sample, we can say the majority of the K-12 students are from U.S. public schools, while the higher education products contain a mix of students from community colleges and four-year institutions, again mainly from the U.S.

To test the retention of the topics after they are mastered, we make use of the ALEKS *progress assessment*, an assessment given at regular intervals that is focused on the student's recent learning. The progress assessment plays a key role in the ALEKS system, as it enforces two learning strategies that have been shown to help with the retention of knowledge: spaced practice [30, 64] and retrieval practice [5, 31, 48, 49, 50]. To evaluate student knowledge retention, we define the *retention rate*—or, the *correct answer rate*—to be the proportion of the time that students answer topics correctly in the progress assessment after having mastered the topics in the ALEKS system.

Our analysis is complicated by a selection bias that exists with the assignment of the different mastery thresholds. That is, the topics using the low mastery threshold, being from the uncertain category, are the ones for which the ALEKS assessment was not confident enough to classify as either known or unknown by the student—as such, it stands to reason that some proportion of these topics are likely known by the students, or that, at the very least, these topics tend to be easier for the students to learn. In comparison, the topics that are classified as unknown by the ALEKS system are typically more difficult for the students.

Thus, to compensate for this issue, we apply the elements of a regression discontinuity design (RDD) [59] to our analysis. RDD is a popular quasi-experimental design that is commonly used in fields such as econometrics [3] and political science [23]. The idea is that, given an experimental condition assigned by an arbitrary cutoff, it's plausible the data points close to this cutoff are similar, regardless of which side of the cutoff they ultimately fall on. In this study, we leverage the fact that a probability cutoff determines the assignment of the mastery threshold in the ALEKS system, with topics below the cutoff being assigned the high mastery threshold, while topics above the cutoff are assigned the low mastery threshold. By comparing topics with probabilities close to the cutoff, we hope to get accurate estimates of the differences between the two mastery thresholds.

In order to apply these ideas, we must account for the fact

that the ALEKS system also uses the information from the prerequisite relationships to assign the mastery threshold to topics. For example, suppose topic x is a prerequisite for topic y . Because of this relationship, if x is answered incorrectly during an ALEKS assessment, topic y will most likely be classified as unknown and thus given the high mastery threshold. Since this decision does not depend directly on the probability cutoff, we exclude these examples from all of our subsequent analyses, so that the probability cutoff is the sole determining factor in assigning the mastery threshold.

4. ANALYZING LEARNING RATIOS

Our first analysis attempts to quantify the differences between the mastery thresholds by comparing the *learning ratios*—that is, the proportions of topics worked on by students that are eventually mastered. As just discussed, we want to only look at data points which have the mastery threshold determined exclusively by the probabilities. Additionally, we restrict our analysis to learning data prior to a student’s first progress assessment, as this assessment can alter the mastery threshold assigned to a topic, and we only select data points for which some work has taken place—for the latter, we minimally require that the student has at least looked at an example instance of the topic. Finally, out of the 2,154 total topics in our data set, we remove 7 that, due to technical issues, have systematic problems with their probability estimates. This leaves us with a total of 58,891,970 data points from 2,181,646 unique students.

Our next step is to select a reasonable *bandwidth* to conduct the RDD analysis, that is, a narrow interval around the probability cutoff to which the data points will be further restricted. While, all else equal, we want to have as much data as possible to work with, we also want our bandwidth to be narrow enough so that the included topics are expected to be similar in difficulty. We choose a bandwidth of 0.02 around the cutoff, which we believe works reasonably well at balancing these competing concerns. This leaves us with 1,949,102 data points from 984,138 unique students.

Table 1: Comparison of outcomes for the low mastery and high mastery groups.³

Mastery threshold	Learn	Fail	Inc.	No resp.
High (956,260)	0.847	0.057	0.062	0.034
Low (992,842)	0.865	0.052	0.050	0.033

For these data points, Table 1 shows the summary statistics after partitioning the learning outcomes into the following four categories.

- Learn: topic successfully mastered
- Fail: topic failed by submitting five consecutive incorrect answers
- Incomplete: at least one answer submitted, but topic is neither learned nor failed
- No response: an instance of the topic is viewed—and possibly an explanation page, as well—but no answers are submitted

³Based on 10,000 cluster bootstrap samples—with the data from each student representing a single “cluster”—the 95% confidence intervals for the point estimates in Table 1 are all less than 0.002 in width.

Based on this partitioning, the learning ratio is simply the proportion of the outcomes in the Learn category. From Table 1 we can see that the topics in the high mastery group have a lower learning ratio in comparison to the low mastery topics—0.847 vs. 0.865, respectively. Furthermore, the high mastery group has larger proportions of incomplete and failed topics. Note that all of these results make intuitive sense—that is, all else being equal, for a given topic we expect it to be harder to learn under the high mastery threshold in comparison to the low mastery threshold.

One concern we have is that students may be actively seeking out one mastery threshold or the other, with perhaps the most prominent worry being that students would try to find the topics with the low mastery threshold, as this information is available to them. While our previous experience working with the ALEKS system has shown us that students mostly work on the specific topic the system presents to them, this is still worth investigating. As a start, we can look at the proportions in the No resp. column of Table 1—here, it’s reassuring that these values are similar for the two different mastery thresholds.

Next, we can look at a *density* plot of the probabilities to see if there is an abrupt change as we move across the probability cutoff. Partitioning the interval $[-0.02, 0.02]$ into 100 bins of width 0.0004 each, in Figure 1 we plot the relative frequency (proportion) vs. the average distance from the threshold, based on the probabilities in each bin. While there’s an increasing trend in the density as the x -values increase—which is a reflection of the distribution of the probabilities, rather than any particular student behavior—we are specifically interested in what happens around the probability cutoff, which is at 0 on the x -axis. As there doesn’t appear to be clear evidence of a discontinuity—i.e., an abrupt increase or decrease—around the cutoff, we can use a *density test* to more precisely check for such a change [41]. Specifically, we apply the procedure outlined in [11], where the null hypothesis assumes there is no discontinuity in the density around the cutoff. Using the R implementation of this procedure in the `rddensity` package [12], the resulting p -value is 0.61—thus, the null hypothesis of no discontinuity is not rejected. Taking these results together, conservatively we can at least say there are no obvious signs of a bias from students electing to work on topics based on the mastery threshold.

5. FORGETTING AND RETENTION

In this next section, we attempt to estimate the differences in retention and forgetting between the topics that have been learned with the two mastery thresholds. While doing so, there are two important factors to consider. First, the students who learn with the high mastery threshold get more practice, as they tend to answer more questions in comparison to those who learn with the low mastery threshold. Second, as we saw previously the high mastery threshold is associated with a lower learning ratio in comparison to the low mastery threshold. This indicates there is a selection bias when we look at the students who learn a topic with the high threshold and compare them to students who learn with the low threshold—that is, the students who pass the high threshold tend to have a slightly better grasp of the material, or are perhaps slightly stronger students. Note that we’d expect both of these factors to benefit the knowledge

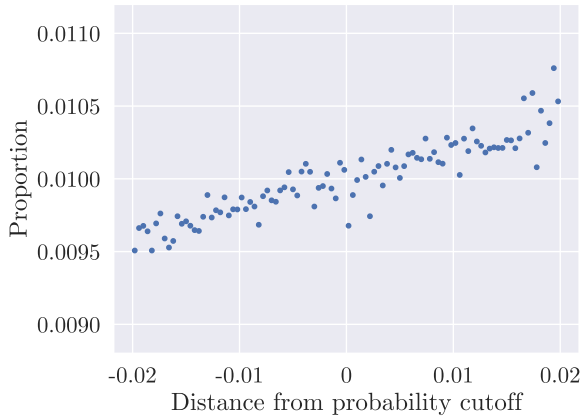


Figure 1: Density plot of the probability values, based on the distance from the probability cutoff. (More precisely, the x -axis reports the difference $\text{topic probability} - \text{cutoff probability}$.)

retention of the students who learn with the high threshold.

While it would be of interest to isolate these factors as much as possible, we won't be able to separate them in the analysis that follows, as they are directly connected to the specific mastery threshold used. On the one hand, from a purely scientific perspective this is unfortunate, as it would be of interest to, say, precisely compare the associations between the different amounts of practice. On the other hand, from a more practical viewpoint we can still analyze the overall differences between the mastery thresholds, which is arguably of more value for designing and improving adaptive learning and intelligent tutoring systems.

Table 2: Average number of actions per learning sequence—numbers in parentheses show the relative proportion of each action, based on the average total number of actions in the bottom row.

	High mastery (278,126)	Low mastery (302,191)
Correct answers	3.6 (0.66)	2.3 (0.61)
Wrong answers	1.3 (0.23)	1.0 (0.27)
Explanations	0.6 (0.10)	0.5 (0.12)
Total	5.4	3.8

Starting with the data set summarized in Table 1, we extract the subset of data points that (a) are successfully learned and (b) appear as questions in the student's first progress assessment—this leaves us with 580,317 data points from 436,735 unique students. In Table 2 we show the learning sequence statistics partitioned by the mastery threshold. The high mastery threshold topics have about 1.6 more learning events, with about 1.5 of these being either correct or wrong answers. Next, in Figure 2 we show a plot of the retention rates based on the distance from the probability cutoff, with the data points being divided into equal-width bins of 0.005, starting at -0.02 and ending at 0.02. For each bin the y -value represents the average correct answer rate when a topic appears in a student's first progress assessment, while the x -value is the average distance from the probability cut-

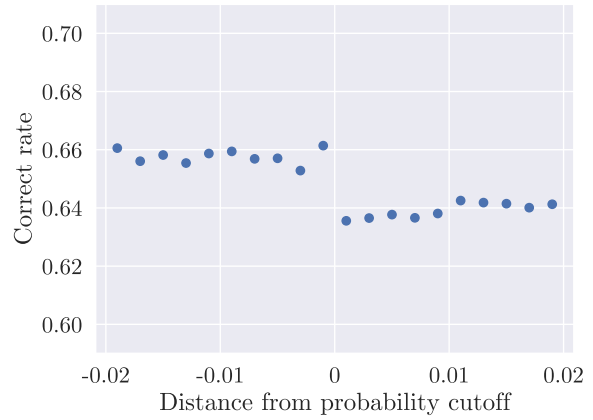


Figure 2: Retention (correct) rates based on the distance from the probability cutoff.

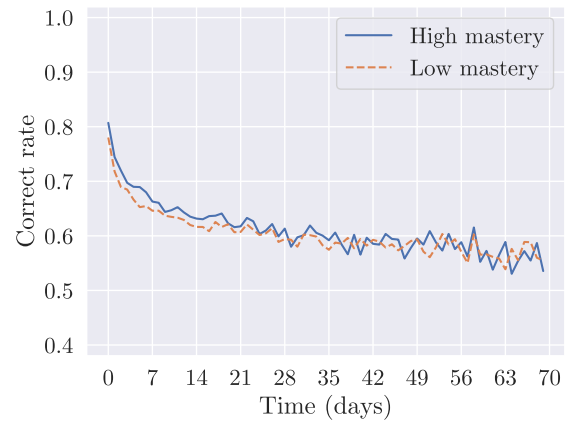


Figure 3: Mastery threshold forgetting curves.

off. Note there is a clear drop in retention as we move across the probability cutoff to the low mastery threshold topics. However, as we are studying the retention of knowledge, one factor we haven't taken into account is time. In particular, we next look at the forgetting curves for these data to see how these relationships might change at different time scales.

To generate these curves, for each data point we compute the time in days between the learning of the topic and its appearance on the progress assessment. Then, we group the data points into bins of width one day, compute the retention rate within each bin, and plot the results in Figure 3. The solid (blue) line shows the curve for the high mastery threshold topics, while the dashed (orange) line shows the curve for the low mastery threshold topics. For time values less than two weeks, the retention rate for the high mastery threshold group is higher—however, for larger time values it's not quite as clear how much of a difference, if any, exists between the retention curves.

We next use a linear regression model to more precisely estimate the differences in retention between the two mastery threshold groups—as our outcome variable is binary, this

model is known as a linear probability model. While the use of a generalized linear model—such as logistic regression—is typically recommended with binary outcome variables, we prefer to use a linear regression here to make it easier for us to interpret the coefficients. Although the use of a linear model with a binary outcome variable could theoretically lead to biased estimates, it’s been argued that this bias is typically low [3]. Additionally, a criticism of the linear probability model is that it could give invalid probability estimates less than zero or greater than one. However, based on previous works analyzing forgetting in the ALEKS system [18, 36, 37, 38], we expect the probability estimates of a correct answer to be bounded well away from zero and one.

As some students appear multiple times within our data, we treat data points associated to the same student as a “group” or “cluster”—this leaves us with 436,735 clusters, one for each unique student. To handle these clusters appropriately, in each of our analyses we fit a marginal model using the generalized estimating equation (GEE) class in the `statsmodels` [54] Python library. GEE models are commonly applied in epidemiological studies and analyses containing repeated measurements [27, 28, 33, 57], making them well-suited for our study.

Our regression models include the following predictors.

- x_1 : 1 for high mastery; 0 for low mastery
- x_2 : Initial assessment probability estimate
- x_3 : Initial assessment score = (number of topics classified as known) / (total number of topics in course)
- x_4 : Categorical variable encoding ALEKS product
- x_5 : Categorical variable encoding first event in learning sequence (correct, incorrect, or explanation)
- x_6 : Categorical variable encoding time (in weeks) since topic was learned (see Table 3)
- x_7 : Interaction between mastery and time ($x_1 \times x_6$)

The variables x_1 and x_7 are our main focus, as we want to estimate the average difference in retention between the two mastery groups—additionally, we want to see if these differences vary across the categories of the time variable. The remaining predictors are control variables, as they can help adjust for factors such as variation in starting knowledge (x_3), general differences between students using the various ALEKS products (x_4), and the amount of initial struggle experienced by the students while learning the topics (x_5). Finally, it’s generally considered good practice to include the assignment variable, represented here by the probability estimate x_2 , in the regression as well [23].

Regarding the time since the topic was learned, a complication with this variable is that it’s technically a *post-treatment* variable—that is, it’s measured after the “treatment” occurs, where in our case the treatment corresponds to the successful learning of the topic with the high mastery threshold. If a causal link is suspected between the post-treatment variable and the treatment, including the post-treatment variable in the regression could bias the estimate of the coefficient for the treatment variable [1, 53]. While we don’t have a compelling reason to think there is a causal link between the

Table 3: Categorical variable for time (x_6).

Category	Description
1	Less than 7 days after learning
2	Between 7 and 14 days after learning
⋮	
9	Between 56 and 63 days after learning
10	More than 63 days after learning

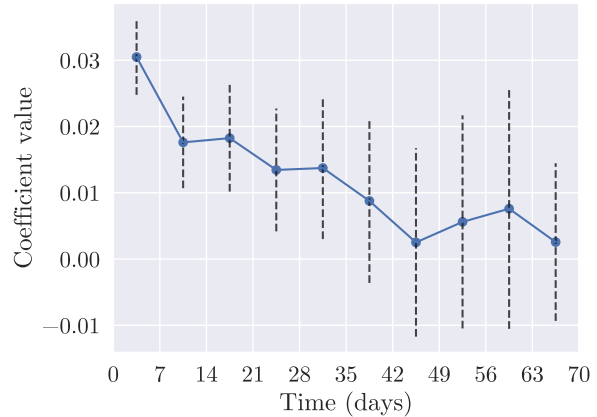


Figure 4: Coefficient estimates of the retention rate differences.

mastery threshold and the time variable, we use the following procedure in an attempt to address the possibility of post-treatment bias. First, we run our regression analysis including the categorical variable for time. Next, we re-run our analysis using the two-step regression procedure known as the *sequential g-estimator* [29, 60]. Using this procedure allows us to make an estimate of β , the coefficient of the treatment, that adjusts for possible bias from the inclusion of the post-treatment variable [1, 24, 29, 60, 61]. Comparing the results using the sequential g-estimator to our first regression, we do not see any substantial differences—for example, in the first model we fit, the estimates of the coefficients of interest differ by less than 0.0013 in absolute value. Thus, to simplify the exposition, in what follows we describe and report the results from the models fit without using the sequential g-estimator.

Figure 4 shows the results from fitting a model with variables x_1 through x_7 . For the given time category, each (blue) dot represents the estimated average difference in retention between the two mastery thresholds, with the dashed lines showing the 95% confidence interval for each estimate. For example, the first dot represents the data points with a retention time of less than seven days, where the high mastery group has an estimated average retention rate that’s higher by about 0.03, with a 95% confidence interval of (0.025, 0.036). The general trend suggests that larger time values are associated with smaller retention differences between the two groups. These results appear to be consistent with the plots shown in Figure 3, where the gap between the two forgetting curves is smaller for the larger time values.

Next, we take a different approach and use a type of matched

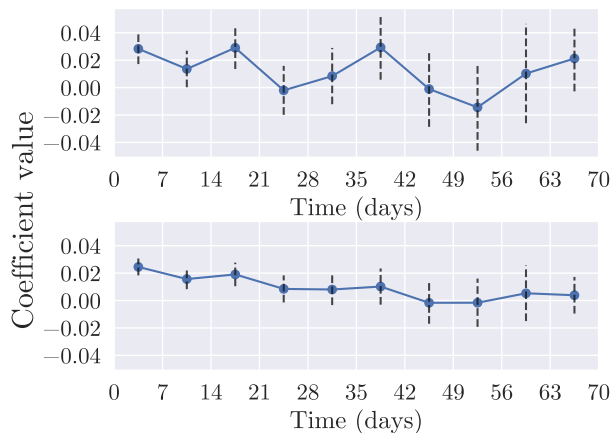


Figure 5: Matched data coefficient estimates, using bandwidths of 0.02 (top) and 0.04 (bottom).

design. Starting with the 580,317 data points from our previous analysis, we restrict the data to the 59,689 students who have learned topics using both mastery thresholds. This leaves us with 72,279 high mastery topics and 73,851 low mastery topics. The results from fitting a model using the variables x_1 through x_7 are shown in the top plot of Figure 5. Similar to the results from the unmatched data in Figure 4, the estimated average difference is about 0.03 for week 1, while then dropping below 0.02 for week 2. While the point estimates appear to have a negative trend—that is, the higher time values are associated with smaller estimated retention rate differences between the two mastery thresholds, on average—the trend is not nearly as pronounced as the one in Figure 4, and the estimates tend to be noisier.

In an attempt to get cleaner estimates of the average retention differences between the two mastery groups, we enlarge our data set by using a wider bandwidth of 0.04—this gives a total of 1,163,706 data points. Restricting our analysis to the 168,339 students who have examples of topics learned with both mastery thresholds, we have a new data set consisting of 476,105 data points. The results from the regression fit on this new data set are shown in the bottom plot of Figure 5, where it’s instructive to see that there is a clearer pattern than in the top plot—that is, using the enlarged data set, there appears to be a fairly strong negative association between the time values and the estimated differences in retention, similar to the results shown in Figure 4.

6. DISCUSSION

In this work we presented a detailed comparison of two mastery learning thresholds that are used in the ALEKS system. Attempting to adjust for selection effects and other possible confounding variables, we utilized elements of a regression discontinuity design by leveraging the fact that the assignment of the mastery threshold is determined by a probability cutoff value. Focusing on topics with probabilities close to this cutoff, we looked at the learning outcomes for the two different mastery threshold groups, with the results suggesting that, while differences do exist, they are not particularly large. For example, the average learning ratio difference between the two groups was less than 0.02. Additionally,

we used regression models to estimate the average difference in knowledge retention between the mastery threshold groups. The overall retention rates were more similar than we might have expected a priori and, furthermore, we saw evidence that the difference in retention rates between the two groups was negatively associated with time—that is, longer time gaps between the initial learning and the test of retention had smaller average differences in retention.

While performing our analyses, we investigated, and attempted to adjust for, several potential sources of bias and confounding. Nonetheless, being an observational study it’s possible that other sources of bias exist. Thus, in what follows we interpret our results within the larger literature on learning and retention, and also discuss potential implications for the ALEKS system, all while keeping this caveat in mind. To start, given that the difference in retention between the mastery threshold groups was smaller for larger time values, this might suggest that any possible gains from the high mastery threshold are not persistent. Notably, prior research on learning has shown that massed practice (i.e., grouping learning into a single session) and overlearning (i.e., continuing to practice after a skill has been mastered), while possibly beneficial in the short-term, do not lead to learning that is especially durable or long lasting [30, 51]. However, as most experiments studying these learning strategies involve simple verbal tasks in a laboratory setting [10, 15, 51], we found it informative to see similar results for students learning mathematics in an adaptive learning system.

Furthermore, while only a limited number of experiments investigate these issues for learning mathematics, two particular studies seem relevant and informative for our current work. First, the results in [51] indicated that the gains from massed practice of mathematics problems did not appear to be as durable as those from using distributed practice—specifically, while the benefits from these techniques appeared similar after a week, with a longer gap of four weeks distributed practice was superior. Second, in [52] two massed practice conditions for learning mathematics problems—with these conditions being somewhat analogous to our high mastery and low mastery conditions—were compared, with no clear difference in performance observed between the conditions. Thus, the outcomes of these two studies are seemingly consistent with the results from our current work.

If the results of this study prove to be valid, a possible adjustment to the ALEKS system is to reduce the usage of the higher mastery threshold for topics close to the probability cutoff. The benefit of this approach is that it would allow students to more efficiently learn additional topics. Taking a slightly different view, it’s well-documented that distributed practice is more effective as an overall learning strategy in comparison to massed practice [10, 15, 20, 30, 64]. Thus, rather than removing the high mastery threshold completely, perhaps the extra practice enforced by the high mastery threshold could be distributed over multiple learning sessions. Additionally, as previous work found evidence that retrieval practice within the ALEKS system is associated with better retention [38], it would be of interest to find the most effective way of combining the principles of both retrieval and distributed practice in the system. This is a line of research we are currently exploring in more detail.

7. REFERENCES

- [1] A. Acharya, M. Blackwell, and M. Sen. Explaining causal findings without bias: Detecting and assessing direct effects. *The American Political Science Review*, 110(3):512–529, 2016.
- [2] P. K. Agarwal, P. M. Bain, and R. W. Chamberlain. The value of applied research: Retrieval practice improves classroom learning and recommendations from a teacher, a principal, and a scientist. *Educational Psychology Review*, 24:437–448, 2012.
- [3] J. D. Angrist and J.-S. Pischke. *Mostly Harmless Econometrics*. Princeton University Press, 2008.
- [4] L. Averell and A. Heathcote. The form of the forgetting curve and the fate of memories. *Journal of Mathematical Psychology*, 55:25–35, 2011.
- [5] C. L. Bae, D. J. Therriault, and J. L. Redifer. Investigating the testing effect: Retrieval as a characteristic of effective study strategies. *Learning and Instruction*, 60:206–214, 2019.
- [6] R. S. J. d. Baker, A. T. Corbett, and V. Alevan. More accurate student modeling through contextual estimation of slip and guess probabilities in Bayesian Knowledge Tracing. In *Intelligent Tutoring Systems*, pages 406–415. Springer Berlin Heidelberg, 2008.
- [7] O. Bälter, D. Zimmaro, and C. Thille. Estimating the minimum number of opportunities needed for all students to achieve predicted mastery. *Smart Learning Environments*, 5(1):1–19, 2018.
- [8] K. Barzagar Nazari and M. Ebersbach. Distributing mathematical practice of third and seventh graders: Applicability of the spacing effect in the classroom. *Applied Cognitive Psychology*, 33(2):288–298, 2019.
- [9] B. S. Bloom. Learning for mastery. *Evaluation Comment*, 1(2), 1968.
- [10] S. K. Carpenter, N. J. Cepeda, D. Rohrer, S. H. Kang, and H. Pashler. Using spacing to enhance diverse forms of learning: Review of recent research and implications for instruction. *Educational Psychology Review*, 24(3):369–378, 2012.
- [11] M. D. Cattaneo, M. Jansson, and X. Ma. Simple local polynomial density estimators. *Journal of the American Statistical Association*, 115(531):1449–1455, 2020.
- [12] M. D. Cattaneo, M. Jansson, and X. Ma. *rddensity: Manipulation Testing Based on Density Discontinuity*, 2021. R package version 2.2.
- [13] H. Cen, K. Koedinger, and B. Junker. Learning Factors Analysis—a general method for cognitive model evaluation and improvement. In *International Conference on Intelligent Tutoring Systems*, pages 164–175. Springer, 2006.
- [14] H. Cen, K. R. Koedinger, and B. Junker. Is over practice necessary?—Improving learning efficiency with the cognitive tutor through educational data mining. In *International Conference on Artificial Intelligence in Education*. IOS Press, 2007.
- [15] N. J. Cepeda, H. Pashler, E. Vul, J. T. Wixted, and D. Rohrer. Distributed practice in verbal recall tasks: A review and quantitative synthesis. *Psychological bulletin*, 132(3):354–380, 2006.
- [16] B. Choffin, F. Popineau, Y. Bourda, and J.-J. Vie. DAS3H: Modeling student learning and forgetting for optimally scheduling distributed practice of skills. In *Proceedings of the 12th International Conference on Educational Data Mining*, pages 29–38, 2019.
- [17] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction*, 4(4):253–278, 1994.
- [18] E. Cosyn, H. Uzun, C. Doble, and J. Matayoshi. A practical perspective on knowledge space theory: ALEKS and its data. *Journal of Mathematical Psychology*, 101:102512, 2021.
- [19] S. Doroudi. Mastery learning heuristics and their hidden models. In *International Conference on Artificial Intelligence in Education*, pages 86–91. Springer International Publishing, 2020.
- [20] J. Dunlosky, K. A. Rawson, E. J. Marsh, M. J. Nathan, and D. T. Willingham. Improving students learning with effective learning techniques: Promising directions from cognitive and educational psychology. *Psychological Science in the Public Interest*, 14(1):4–58, 2013.
- [21] H. Ebbinghaus. *Memory: A Contribution to Experimental Psychology*. Originally published by Teachers College, Columbia University, New York, 1885; translated by Henry A. Ruger and Clara E. Bussenius (1913).
- [22] S. Fancsali, T. Nixon, and S. Ritter. Optimal and worst-case performance of mastery learning assessment with Bayesian knowledge tracing. In *Proceedings of the Sixth International Conference on Educational Data Mining*, pages 35–42, 2013.
- [23] A. Gelman, J. Hill, and A. Vehtari. *Regression and Other Stories*. Cambridge University Press, 2020.
- [24] S. Goetgeluk, S. Vansteelandt, and E. Goetghebeur. Estimation of controlled direct effects. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(5):1049–1066, 2008.
- [25] N. A. Goossens, G. Camp, P. P. Verkoeijen, H. K. Tabbers, S. Bouwmeester, and R. A. Zwaan. Distributed practice and retrieval practice in primary school vocabulary learning: A multi-classroom study. *Applied Cognitive Psychology*, 30(5):700–712, 2016.
- [26] P. Hanley-Dunn and J. L. McIntosh. Meaningfulness and recall of names by young and old adults. *Journal of Gerontology*, 39:583–585, 1984.
- [27] J. W. Hardin and J. M. Hilbe. *Generalized Estimating Equations*. Chapman and Hall/CRC, 2012.
- [28] P. J. Heagerty and S. L. Zeger. Marginalized multilevel models and likelihood inference (with comments and a rejoinder by the authors). *Statistical Science*, 15(1):1–26, 2000.
- [29] M. M. Joffe and T. Greene. Related causal frameworks for surrogate outcomes. *Biometrics*, 65(2):530–538, 2009.
- [30] S. H. Kang. Spaced repetition promotes efficient and effective learning: Policy implications for instruction. *Policy Insights from the Behavioral and Brain Sciences*, 3(1):12–19, 2016.
- [31] J. D. Karpicke and H. L. Roediger. The critical importance of retrieval for learning. *Science*, 319(5865):966–968, 2008.

- [32] K. Kelly, Y. Wang, T. Thompson, and N. Heffernan. Defining mastery: Knowledge tracing versus n-consecutive correct responses. pages 630–631, 2015.
- [33] K.-Y. Liang and S. L. Zeger. Longitudinal data analysis using generalized linear models. *Biometrika*, 73(1):13–22, 1986.
- [34] R. V. Lindsey, J. D. Shroyer, H. Pashler, and M. C. Mozer. Improving students long-term knowledge retention through personalized review. *Psychological science*, 25(3):639–647, 2014.
- [35] J. Matayoshi, E. Cosyn, and H. Uzun. Evaluating the impact of research-based updates to an adaptive learning system. In *International Conference on Artificial Intelligence in Education*, pages 451–456. Springer, 2021.
- [36] J. Matayoshi, U. Granziol, C. Doble, H. Uzun, and E. Cosyn. Forgetting curves and testing effect in an adaptive learning and assessment system. In *Proceedings of the 11th International Conference on Educational Data Mining*, pages 607–612, 2018.
- [37] J. Matayoshi, H. Uzun, and E. Cosyn. Deep (un)learning: Using neural networks to model retention and forgetting in an adaptive learning system. In *International Conference on Artificial Intelligence in Education*, pages 258–269, 2019.
- [38] J. Matayoshi, H. Uzun, and E. Cosyn. Studying retrieval practice in an intelligent tutoring system. In *Proceedings of the Seventh ACM Conference on Learning @ Scale*, pages 51–62, 2020.
- [39] J. Matayoshi, H. Uzun, and E. Cosyn. Using a randomized experiment to compare the performance of two adaptive assessment engines. To appear in *Proceedings of the 15th International Conference on Educational Data Mining*, 2022.
- [40] D. M. McBride and B. A. Doshier. A comparison of forgetting in an implicit and explicit memory task. *Journal of Experimental Psychology: General*, 126:371–392, 1997.
- [41] J. McCrary. Manipulation of the running variable in the regression discontinuity design: A density test. *Journal of econometrics*, 142(2):698–714, 2008.
- [42] A. Paivio and P. C. Smythe. Word imagery, frequency, and meaningfulness in short-term memory. *Psychonomic Science*, 22:333–335, 1971.
- [43] Z. A. Pardos and N. T. Heffernan. KT-IDEM: Introducing item difficulty to the knowledge tracing model. In *Proceedings of the 19th International Conference on User Modeling, Adaption, and Personalization*, UMAP’11, pages 243–254. Springer-Verlag, 2011.
- [44] P. I. Pavlik and J. R. Anderson. Using a model to compute the optimal schedule of practice. *Journal of Experimental Psychology: Applied*, 14(2):101–117, 2008.
- [45] P. I. Pavlik, H. Cen, and K. R. Koedinger. Performance Factors Analysis—a new alternative to knowledge tracing. In *International Conference on Artificial Intelligence in Education*, pages 531–538, 2009.
- [46] R. Pelánek and J. Řihák. Experimental analysis of mastery learning criteria. In *Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization*, pages 156–163, 2017.
- [47] Y. Qiu, Y. Qi, H. Lu, Z. A. Pardos, and N. T. Heffernan. Does time matter? modeling the effect of time with Bayesian knowledge tracing. In *Proceedings of the Fourth International Conference on Educational Data Mining*, pages 139–148, 2011.
- [48] H. L. Roediger III and A. C. Butler. The critical role of retrieval practice in long-term retention. *Trends in Cognitive Sciences*, 15:20–27, 2011.
- [49] H. L. Roediger III and J. D. Karpicke. The power of testing memory: Basic research and implications for educational practice. *Perspectives on Psychological Science*, 1(3):181–210, 2006.
- [50] H. L. Roediger III and J. D. Karpicke. Test-enhanced learning: Taking memory tests improves long-term retention. *Psychological Science*, 17(3):249–255, 2006.
- [51] D. Rohrer and K. Taylor. The effects of overlearning and distributed practice on the retention of mathematics knowledge. *Applied Cognitive Psychology: The Official Journal of the Society for Applied Research in Memory and Cognition*, 20(9):1209–1224, 2006.
- [52] D. Rohrer and K. Taylor. The shuffling of mathematics problems improves learning. *Instructional Science*, 35(6):481–498, 2007.
- [53] P. R. Rosenbaum. The consequences of adjustment for a concomitant variable that has been affected by the treatment. *Journal of the Royal Statistical Society: Series A (General)*, 147(5):656–666, 1984.
- [54] S. Seabold and J. Perktold. Statsmodels: Econometric and statistical modeling with Python. In *9th Python in Science Conference*, 2010.
- [55] B. Settles and B. Meeder. A trainable spaced repetition model for language learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1848–1858, 2016.
- [56] S. M. Smith. Remembering in and out of context. *Journal of Experimental Psychology: Human Learning and Memory*, 4:460–471, 1979.
- [57] C. Szmargd, P. Clarke, and F. Steele. Subject specific and population average models for binary longitudinal data: a tutorial. *Longitudinal and Life Course Studies*, 4(2):147–165, 2013.
- [58] B. Tabibian, U. Upadhyay, A. De, A. Zarezade, B. Schölkopf, and M. Gomez-Rodriguez. Enhancing human learning via spaced repetition optimization. *Proceedings of the National Academy of Sciences*, 116(10):3988–3993, 2019.
- [59] D. L. Thistlethwaite and D. T. Campbell. Regression-discontinuity analysis: An alternative to the ex post facto experiment. *Journal of Educational psychology*, 51(6):309–317, 1960.
- [60] S. Vansteelandt. Estimating direct effects in cohort and case-control studies. *Epidemiology*, 20:851–860, 2009.
- [61] S. Vansteelandt, S. Goetgeluk, S. Lutz, I. Waldman, H. Lyon, E. E. Schadt, S. T. Weiss, and C. Lange. On the adjustment for covariates in genetic association analysis: a novel, simple principle to infer direct causal effects. *Genetic Epidemiology: The Official Publication of the International Genetic Epidemiology*

- Society*, 33(5):394–405, 2009.
- [62] Y. Wang and J. E. Beck. Incorporating factors influencing knowledge retention into a student model. In *Proceedings of the Fifth International Conference on Educational Data Mining*, pages 201–203, 2012.
- [63] Y. Wang and N. T. Heffernan. Towards modeling forgetting and relearning in ITS: Preliminary analysis of ARRS data. In *Proceedings of the Fourth International Conference on Educational Data Mining*, pages 351–352, 2011.
- [64] Y. Weinstein, C. R. Madan, and M. A. Sumeracki. Teaching the science of learning. *Cognitive Research: Principles and Implications*, 3(1):1–17, 2018.
- [65] X. Xiong and J. E. Beck. A study of exploring different schedules of spacing and retrieval interval on mathematics skills in ITS environment. In *International Conference on Intelligent Tutoring Systems*, pages 504–509. Springer, 2014.
- [66] X. Xiong, S. Li, and J. E. Beck. Will you get it right next week: Predict delayed performance in enhanced ITS mastery cycle. In *The Twenty-Sixth International FLAIRS Conference*, 2013.
- [67] X. Xiong, Y. Wang, and J. B. Beck. Improving students’ long-term retention performance: A study on personalized retention schedules. In *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge*, pages 325–329. ACM, 2015.
- [68] M. Yudelson. Individualizing Bayesian knowledge tracing. Are skill parameters more important than student parameters? In *Proceedings of the Ninth International Conference on Educational Data Mining*, pages 556–561, 2016.

Improving Peer Assessment with Graph Neural Networks

Alireza A. Namanloo
Ontario Tech University
Oshawa, Canada
ali@ali-naman.com

Julie Thorpe
Ontario Tech University
Oshawa, Canada
julie.thorpe@uoit.ca

Amirali Salehi-Abari
Ontario Tech University
Oshawa, Canada
abari@uoit.ca

ABSTRACT

Peer assessment systems are emerging in many settings, such as peer grading in large (online) classes, peer review in conferences, peer art evaluation, etc. However, peer assessments might not be as accurate as expert evaluations, thus rendering these systems unreliable. The reliability of peer assessment systems is influenced by various factors such as assessment ability of peers, their strategic assessment behaviors, and the peer assessment setup (e.g., peer evaluating group work or individual work of others). In this work, we first model peer assessment as multi-relational weighted networks that can express a variety of peer assessment setups, and can also capture conflicts of interest and strategic behaviors. Leveraging our peer assessment network model, we introduce a graph neural network which can learn assessment patterns and user behaviors to more accurately predict expert evaluations. Our extensive experiments on real and synthetic datasets demonstrate the efficacy of our approach, which outperforms a variety of peer assessment methods.

Keywords

Peer Assessment, Graph Neural Network.

1. INTRODUCTION

Peer assessment systems have emerged as a cost-effective and scalable evaluation mechanism in many settings such as peer grading in large (online) classes and peer review in conferences. In these systems, peers assess each others' work (e.g., assignments, papers, etc.) in lieu of a set of pre-appointed experts responsible for evaluation (e.g., instructors, teaching assistants, program committee members, etc.). These peer assessment systems not only make the evaluation of thousands of contributions plausible, but also help to deepen peers' understanding [22], and facilitate peers providing feedback to each other [20]. However, the reliability of peer assessment systems is directly impacted by the accuracy of peers in their assessments. Peers might lack knowledge or motivation to accurately evaluate others, or

they might be strategic in their assessments for their own gain [1, 12, 11, 2, 15, 34, 25, 21].

Related work. Two classes of approaches are taken to address reliability challenges. One primarily focuses on designing strategy-proof peer assessment mechanisms, which incentivize peers to accurately assess each other [6, 24, 9, 16, 11, 35, 32]. The other class of approaches—most relevant to our work—emphasizes learning peer aggregation mechanisms, which aggregate noisy peer assessments for an item (e.g., assignment or paper) as an estimate of its ground-truth valuation (or expert evaluation) [19, 27, 7, 30, 4]. The learning methods for peer assessment aggregation fall into unsupervised [27, 19, 5] and semi-supervised [7, 30] approaches based on whether or not a subset of ground-truth labels are used for training in addition to peer assessment data. These models usually possess particular inductive biases such as peer's assessment accuracy being correlated with his/her item's ground-truth valuations (e.g., the grade of his/her assignment) [19, 27, 7]; or peer's accuracy in an assessment depending on the extent of its agreement with others' assessments or ground-truth valuations [30]. However, these machine learning methods are empirically shown to be only as effective as simple aggregation mechanisms such as averaging [23]. Moreover, these approaches are not flexible and general enough to accommodate a wide variety of peer assessment modes (e.g., when an individual assesses the group contribution of others or self assessments). Our focus in this paper is to develop a semi-supervised aggregation mechanism without any specific or restrictive inductive bias, accommodating various modes of peer assessments.

Contribution. We first introduce our graph representation model of peer assessment, which we call *social-ownership-assessment network (SOAN)*.¹ Our SOAN model can express a wide variety of peer assessment setups (e.g., self-assessment and peer assessment for both individual or group contributions) and represent conflict-of-interest relations between peers using auxiliary information, such as social networks. Leveraging our SOAN model, we then introduce a semi-supervised graph convolutional network (GCN) approach, called *GCN-SOAN*, which can learn assessment patterns and behaviors of peers, without any restrictive inductive bias, to predict ground-truth valuations. We run extensive experiments on real-world and synthetic datasets to evaluate the efficacy of GCN-SOAN. Our GCN-SOAN outperforms a wide variety of baseline methods (including sim-

A. A. Namanloo, J. Thorpe, and A. Salehi-Abari. Improving peer assessment with graph neural networks. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 325–332, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853014>

¹SOAN can read as “swan.”

ple heuristics, semi-supervised, and unsupervised approaches) on the same real-world dataset [23], which was shown to be challenging for machine learning approaches. Our GCN-SOAN also outperforms others on a wide range of synthetic data, which captures strategic assessment behavior between users, follows the assumptions of competitor baselines, or considers strict and generous graders. GCN-SOAN can be a stand-alone approach or possibly be integrated with some incentivizing mechanisms (e.g., [32, 5, 35]).

2. PROPOSED APPROACH

Our goal is to predict the ground-truth assessments (e.g., expert evaluations of educational or professional work) from noisy peer assessments. We first discuss our proposed graph representation model, *social-ownership-assessment network (SOAN)*, for capturing the peer grading behavior. We then present a modified graph convolutional network (GCN), which leverages our SOAN model, to predict the ground-truth assessments. We call this approach *GCN-SOAN*.

2.1 Social-Ownership-Assessment Model

We assume that a set of n users \mathcal{U} (e.g., students or scholars) can assess a set of m items \mathcal{I} (e.g., a set of educational, professional, or intellectual work). The examples cover various applications ranging from peer grading in classrooms to peer reviewing scientific papers, professional work, or research grant applications. We also consider each item $i \in \mathcal{I}$ possesses a (possibly unknown, but verifiable) ground-truth value $v_i \in \mathbb{R}^+$ (e.g., staff grade for a course work, or expert evaluation of intellectual or professional work).

The user-item assessments can be represented by *assessment matrix* $\mathbf{A} = [A_{ui}]$, where A_{ui} is the assessment (e.g., grade or rating) of user $u \in \mathcal{U}$ for item $i \in \mathcal{I}$. We let $A_{ui} = 0$ when the user u 's assessment for item i is missing; otherwise $A_{ui} \in \mathbb{R}^+$. As the assessment matrix \mathbf{A} is sparse, we equivalently represent it by an undirected weighted bipartite graph, consisting of two different node types of users \mathcal{U} and items \mathcal{I} , and weighted assessment edges between them (see Figure 1a as an example).

We introduce a *social-ownership-assessment network (SOAN)*, an undirected weighted multigraph, consisting of three types of social, ownership, and assessment relationships on two node types of users and items. In addition to the assessment matrix \mathbf{A} , this network consists of two other adjacency matrices: *social matrix* $\mathbf{S} = [S_{uv}] \in \mathbb{R}^{n \times n}$ and *ownership matrix* $\mathbf{O} = [O_{ui}] \in \mathbb{R}^{n \times m}$. The social matrix \mathbf{S} , by capturing the friendship and foe relationships between users \mathcal{U} , can accommodate "conflict of interest" information. The ownership matrix \mathbf{O} , by capturing which users to what extent own or contributed to an item, not only completes conflict of interest information but also provides flexibility of modeling group contributions, self-evaluation, etc. We let $G = (\mathbf{S}, \mathbf{O}, \mathbf{A})$ denote the tuple of all three networks of SOAN. Figure 1 demonstrates some instantiations of our models for various settings. SOAN offers various advantages over the existing peer assessment models (e.g., [27, 19, 7, 4]):

Expressiveness. Our model is more *expressive* as it facilitates the representation of many various peer assessment settings that could not be accommodated in the existing models. Its expressive power can be realized in the settings such as self

assessments (Figure 1b), peer assessments for both solo and group work (Figures 1e and 1f), and the mixtures of peer and self assessments for solo and group work (Figures 1c and 1d). For all of these settings, our SOAN model can also express conflict of interest (which is neglected in other models) through a social network (see Figure 1g).

Less Assumptions. Dissimilar to some existing models (e.g., [27, 19, 7, 30]), our model avoids making explicit or implicit assumptions about the relationships between ground-truth values (or grades) and the quality of peer assessments. It is still flexible enough to learn such correlations from assessment data if it exists. Our experiments below have shown that our model outperforms other models with restrictive assumptions regardless of whether their assumptions are present in the data or not.

2.2 Graph Convolutional Networks

Our learning task is semi-supervised. Given a social-ownership-assessment network $G = (\mathbf{S}, \mathbf{O}, \mathbf{A})$ and a set of ground-truth valuations $V_{\mathcal{D}} = \{v_j | j \in \mathcal{D}\}$ for a subset of items $\mathcal{D} \subset \mathcal{I}$, we aim to predict v_i for $i \notin \mathcal{D}$. More specifically, we aim to learn the function $f(i|\boldsymbol{\theta}, G)$ for predicting the ground-truth valuation v_i by $\hat{v}_i = f(i|\boldsymbol{\theta}, G)$. The model parameters $\boldsymbol{\theta}$ are learned from both the observed ground-truth valuations $V_{\mathcal{D}} = \{v_j | j \in \mathcal{D}\}$ and social-ownership-assessment network G . We formulate the function f by a modified graph convolution network (GCN) with a logistic head:

$$f(i|\boldsymbol{\theta}, G) = \sigma(\mathbf{w}^{(o)} \mathbf{z}_i + b^{(o)}), \quad (1)$$

where $\sigma(\cdot)$ is the sigmoid function for converting the linear transformation of the node i 's embedding \mathbf{z}_i into its predicted valuations. Here, \mathbf{w}_o and b_o are the weight vector and the bias parameter for the output layer. The node (i.e., item) embedding \mathbf{z}_i is computed with K layers of graph convolution network. Let $\mathbf{H}^{(l)}$ be the $(n+m) \times d$ matrix of d -dimensional node embeddings at layer l for all users \mathcal{U} and items \mathcal{I} such that user u and item i 's vector embeddings are located at the u -th and $(m+i)$ -th rows, respectively. In Eq. 1, the item i 's embedding \mathbf{z}_i is the $(m+i)$ -th row of $\mathbf{H}^{(K)}$ with the updating rule of

$$\mathbf{H}^{(l+1)} = g^{(l)}(\mathbf{D}^{-1} \mathbf{M} \mathbf{H}^{(l)} \mathbf{W}^{(l)}). \quad (2)$$

The matrix \mathbf{M} is constructed from the graph $G = (\mathbf{S}, \mathbf{O}, \mathbf{A})$ by $\mathbf{M} = \begin{pmatrix} \mathbf{S} & \mathbf{P} \\ \mathbf{P}^\top & \mathbf{0}_m \end{pmatrix} + \mathbf{I}$, where $\mathbf{P} = \mathbf{O} + \mathbf{A}$, \top is the transpose operator, $\mathbf{0}_m$ is $m \times m$ zero matrix, and \mathbf{I} is the identity matrix. In Eq. 2, \mathbf{D} is the diagonal matrix with $D_{ii} = \sum_j \mathbb{1}[M_{ij} \neq 0]$ with $\mathbb{1}[\cdot]$ as the indicator function. The core idea in Eq. 2 is to update the node embeddings at layer $l+1$, denoted by $\mathbf{H}^{(l+1)}$, from layer l 's node embeddings $\mathbf{H}^{(l)}$. This update includes the multiplication of the layer l 's embeddings $\mathbf{H}^{(l)}$ by the normalized matrix $\mathbf{D}^{-1} \mathbf{M}$, then linear transformation by learned weight matrix $\mathbf{W}^{(l)}$ at layer l , and finally passing through a non-linear activation function $g^{(l)}$. The initial embedding matrix $\mathbf{H}^{(0)}$ can be node-level features (e.g., textual features for items, user profiles for users, etc.). When the node-level features are absent, the common practice is to initialize the embeddings with one-hot indicators [14, 28, 10]. As our GCN is built upon SOAN, we refer to this combination as *GCN-SOAN*.

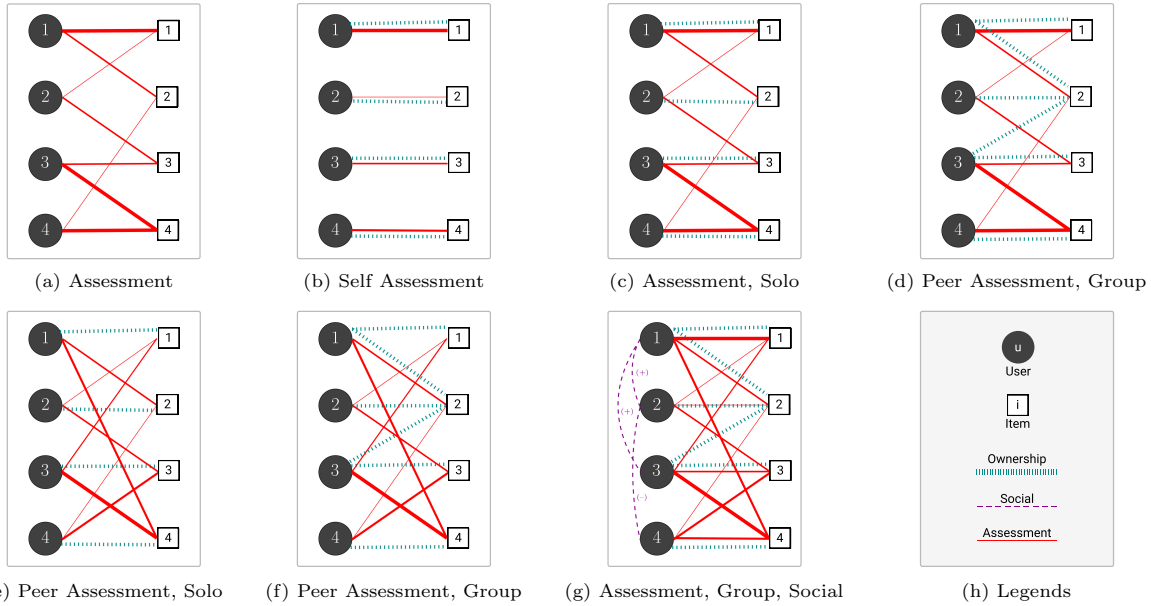


Figure 1: Different instantiations of social-ownership-assessment network (SOAN): (a) assessments provided by users to items as weighted edges; thicker lines for higher weights; (b) self assessments of users for their own items; (c) combination of self and peer assessments of solo contributions; (d) self and peer assessments of both solo and group contributions; (e) peer assessments of solo contributions; and (g) self and peer assessments of group contributions with the social networks between users for capturing conflict of interest.

The updating rule in GCN-SOAN (see Eq. 2) benefits from row normalization of the adjacency matrix similar to many other graph neural networks [31, 26, 28, 29]. As the choice of an effective normalization technique is an application-specific question [10, 3], we have decided to normalize our weighted SOAN model by taking an unweighted average, which has been suggested as a solution to address the sensitivity to node degrees for neighborhood normalization [10].

Our GCN-SOAN differentiates from vanilla GCN in various ways: (i) GCN-SOAN supports weighted graphs as opposed to GCN which solely is designed for unweighted graphs; (ii) it has asymmetric normalization as opposed symmetric normalization. These properties well-equip our GCN-SOAN to aggregate the information from multi-hop neighborhoods (e.g., neighbors, neighbors of neighbors, and so on) of SOAN, thus successfully capturing various assessment behaviors and patterns as evidenced in our experiments below.

Richer data as node-level features. To incorporate richer data for users (e.g., grader’s profile, reviewer’s expertise level, reviewer’s interest, etc.) and items (e.g., textual or visual information) in peer assessment systems, our GCN-SOAN can readily accommodate those information in the form of their node-level features. For example, the expertise level of peers can be represented as the one-hot encoding for initial embeddings. These initial embeddings can be extended with any other type of peers’ auxiliary information (e.g., education, age, sex, etc.). Similarly, initial embeddings of items can accommodate item features (e.g., the keywords for papers, textual features extracted from a paper, etc.).

Learning. Given the SOAN of G and a small training set of ground-truth valuations $V_{\mathcal{D}}$, we learn GCN-SOAN parameters by minimizing the *mean square error* of its predictions:

$$L(\theta|G, \mathcal{D}) = \frac{1}{|\mathcal{D}|} \sum_{i=1}^{|\mathcal{D}|} (v_i - f(i|\theta, G))^2, \text{ where } |\mathcal{D}| \text{ is the number of items in the training dataset, and } f(i|\theta, G) \text{ is the estimated valuation of GCN-SOAN for item } i. \text{ This loss function can be minimized by gradient-based optimization techniques (e.g., stochastic gradient descent, Adam, etc.).}$$

As opposed to many existing peer assessment systems with unsupervised learning approaches (e.g., [19, 4, 27]), we deliberately have adopted a semi-supervised learning approach for predicting ground-truth assessment. This choice offers many advantages at some cost of access to a small training dataset. By learning from the training data, GCN-SOAN is well-equipped to mitigate the influence of strategic behaviors, assessment biases, and unreliable assessments in peer assessment systems. Of course, the extent of this mitigation depends on the size of training data.

3. EXPERIMENTS

We run extensive experiments on real-world and synthetic datasets to compare our GCN-SOAN model against other peer assessment methods. While the real-world datasets allow us to assess the practical efficacy of our approach, we generate various synthetic data to assess its robustness in various settings (e.g., strategic and biased assessments).²

Real-world dataset. The peer grading datasets of Sajjadi et al. [23] includes peer and self grades of 219 students for exercises (i.e., questions) of four assignments and their ground-truth grades.³ For each specific assignment, the sub-

²See the longer version for additional experiments [17].

³The datasets can be found at http://www.tml.cs.uni-tuebingen.de/team/luxburg/code_and_data/peer_grading_data_request_new.php. The original datasets are for six assignments. However, two of the datasets have ordinal peer gradings, not applicable to our experiments.

Table 1: The summary statistics of real-world peer grading datasets.

Asst. ID	Average Grades			Number of					
	Ground-truth	Peer	Self	Exercises	Groups	Students	Items	Peer grades	Self grades
1	0.62 ± 0.27	0.70 ± 0.26	0.74 ± 0.22	3	75	183	225	965	469
2	0.71 ± 0.24	0.76 ± 0.23	0.80 ± 0.22	4	77	206	308	1620	755
3	0.69 ± 0.33	0.75 ± 0.31	0.82 ± 0.26	5	76	193	380	1889	890
4	0.59 ± 0.27	0.68 ± 0.29	0.76 ± 0.24	3	79	191	237	1133	531

missions are group work of 1–3 students, but each student individually has self and peer graded all exercises of two other submissions (in a double-blind setup). We treat all data associated with each assignment as a separate dataset, where all the submitted solutions to its exercises form the item set \mathcal{A} and the user set \mathcal{U} includes all students who have been part of a submission. We also have scaled peer, self, and ground-truth grades to be in the range of $[0, 1]$. Table 1 shows the statistics summary of our datasets.⁴

Synthetic datasets. We discuss different models used for the generation of our synthetic data.

Ground-truth valuation/grade generation. For all $i \in \mathcal{A}$, we sample the true valuation v_i from a mixture of two normal distributions $v_i \sim P(x; \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\sigma}) = \sum_{c=1}^2 \pi_c \mathcal{N}(x; \mu_c, \sigma_c)$, where $\boldsymbol{\pi} = (\pi_1, \pi_2)$, $\boldsymbol{\mu} = (\mu_1, \mu_2)$, and $\boldsymbol{\sigma} = (\sigma_1, \sigma_2)$, with π_k , μ_k , and σ_k being the mixing probability, mean, and standard deviation of the component k .

Social network generation. We create social networks between users by *Erdős-Rényi (ER)* random graph $G(n, p)$ model: each pair of n users are connected to each other with the connection probability of p .

Ownership network generation. For all synthetic datasets, we randomly connect each user to just one item (i.e., one-to-one correspondence between users and items). This setup is in favor of existing peer assessment methods (e.g., [19, 27, 7]), which does not support group ownerships of items.

Assessment network (or peer grades) generation. To generate peer assessment for each item $i \in \mathcal{A}$, we first randomly select a set of k users $N(i) \subset \mathcal{U}$ such that $|N(i)| = k$. Then, for each $u \in N(i)$, we set u 's assessment for item i , denoted by A_{ui} , using one of these two models. The *strategic* model sets $A_{ui} = 1$, if the grader u is a friend of the user j who owns the item i (i.e., $s_{uj} \cdot o_{ji} = 1$); otherwise A_{ui} comes from a normal distribution with the mean of v_i and standard deviation of σ_H . This implies that friends collude to peer grade each others with the highest grade, but would be relatively fair and reliable in assessing a “stranger.” The *bias-reliability* model draws A_{ui} from the normal distribution $\mathcal{N}(x; \hat{\mu}, \hat{\sigma})$ with the mean $\hat{\mu} = v_i + \alpha$ and $\hat{\sigma} = \sigma_{max}(1 - \beta v_l)$, where v_l is the true valuation of item l owned by user u , and σ_{max} is the maximum possible standard deviation (i.e., unreliability) for peer graders. Here, the *bias* parameter $\alpha \in [-1, 1]$ controls the degree of generosity (for $\alpha > 0$) or strictness (for $\alpha < 0$) of the peer grader. The *reliability* parameter β

⁴While GCN-SOAN can easily accommodate user/item features, we were not able to explore its full potential due to a lack of access to datasets with such features.

controls the extent that the reliability of the grader is correlated with his/her item’s grade (i.e., the peer graders with higher grades are more reliable graders). The inductive bias of many peer assessment models (e.g., [27, 7, 19]) include the assumption that the grader’s reliability is a function of his/her item’s grade. Our bias-reliability model allows us to generate synthetic datasets with the presence of this assumption. So we can compare our less-restrictive GCN-SOAN with those models tailored to this specific assumption in such datasets.

Baselines. We compare the performance of our GCN-SOAN model with PeerRank [27], PG1 [19], RankwithTA [7], Vancouver [4], Average, and Median. Average and Median (resp.) outputs the average and median (resp.) of each item’s peer grades as its predicted evaluation. As PeerRank, PG1, and RankwithTA treat users and items interchangeably, they can’t be directly applied to our real-world data with individual assessments on group submissions. For these methods, we preprocess our real-world dataset by taking the average of the grades provided by a group’s members for a particular submission as the group assessment for the submission. For the PeerRank and PG1, we have used the same parameter settings reported by the original papers. The parameters for RankwithTA and Vancouver are selected by grid search with multiple runs, since the optimal parameters either were not reported or result in non-competitive performance.⁵

Experimental setup. We implement GCN-SOAN based on PyTorch [18] and PyTorch Geometric [8].⁶ For all experiments, we use two GCN-SOAN convolutional layers with an embedding dimension of 64 and ELU as activation functions of all hidden layers. We train the model for 800 epochs with Adam optimizer [13] and a learning rate of 0.02. We initialize the node embeddings with vectors of ones. We use Monte Carlo cross-validation [33] with the training-testing splitting ratio of 1:9 (in synthetic data) or 1:4 (in real-world data), implying that just 10% or 20% data is used for training and the rest for testing. To make our results even more robust, we run all tested methods (our model and baselines) on four random splits and report the average error over those splits. For each random split, we compute the root mean square error (RMSE) over testing data as the prediction error.

Results: Real-World Datasets. To assess the effectiveness of GCN-SOAN in predicting ground-truth valuations, we compare it against the baseline methods on eight real-world

⁵We set the Vancouver’s precision parameter to 0.1. For RankwithTA, we set 0.8 and 0.1 (respectively) for the parameters controlling the impact of working ability on grading ability and grading ability on the grade (respectively).

⁶The implementation of GCN-SOAN can be obtained from: <https://github.com/naman-ali/GCN-SOAN/>

Table 2: Root mean square error of various methods over two classes of real-world datasets. The first and second best are shown with dark and light gray backgrounds, respectively. \uparrow and \downarrow denote better and worse than Average. GCN-SOAN (ours) is the only method that consistently has outperformed Average for all datasets. Results are averaged over three runs.

Model	Peer evaluation				Peer and self evaluation			
	Asst. 1	Asst. 2	Asst. 3	Asst. 4	Asst. 1	Asst. 2	Asst. 3	Asst. 4
Average	0.1917	0.1712	0.1902	0.1989	0.1944	0.1681	0.2023	0.2117
Median	0.1991 \downarrow	0.1843 \downarrow	0.2047 \downarrow	0.2250 \downarrow	0.2111 \downarrow	0.1750 \downarrow	0.2333 \downarrow	0.2538 \downarrow
PeerRank	0.1913 \uparrow	0.1762 \downarrow	0.2235 \downarrow	0.2087 \downarrow	0.1888 \uparrow	0.1721 \downarrow	0.2203 \downarrow	0.2168 \downarrow
PG1	0.1919 \downarrow	0.1669 \uparrow	0.2110 \downarrow	0.2161 \downarrow	0.2009 \downarrow	0.1680 \uparrow	0.2111 \downarrow	0.2304 \downarrow
RankwithTA	0.1922 \downarrow	0.1903 \downarrow	0.2183 \downarrow	0.1740 \uparrow	0.1884 \uparrow	0.1845 \downarrow	0.2137 \downarrow	0.1792 \uparrow
Vancouver	0.1851 \uparrow	0.1688 \uparrow	0.1951 \downarrow	0.2071 \downarrow	0.1815 \uparrow	0.1672 \uparrow	0.1945 \uparrow	0.2101 \uparrow
GCN-SOAN (ours)	0.1795 \uparrow	0.1673 \uparrow	0.1869 \uparrow	0.1822 \uparrow	0.1778 \uparrow	0.1621 \uparrow	0.1840 \uparrow	0.1821 \uparrow

datasets. These datasets differentiate on (i) which assignment dataset is used and (ii) whether both peer and self-grades are used or only peer grades. For GCN-SOAN, we just create an assessment network, thus allowing us to measure how the assessment network alone can improve the predication accuracy. As shown in Table 2, our GCN-SOAN model outperforms others in five datasets and ranked second in remaining ones with a small margin. RankwithTA and PG1 are the only models that slightly outperform GCN-SOAN for those three datasets. Notably, GCN-SOAN is the only model which consistently outperformed the simple Average benchmark. This observation is consistent with Sajjadi et al.’s findings [23] on the same dataset that the existing machine learning methods (not including ours) could not improve results over simple baselines. However, the conclusion does not hold anymore as our machine learning GCN-SOAN approach could consistently improve over simple baselines. This improvement mainly arises from the expressive power and generalizability of GCN-SOAN (discussed in Section 2).

Results: Synthetic Data with Bias-Reliability. We run an extensive set of experiments with the bias-reliability peer grade generation model to assess our GCN-SOAN under various peer assessment settings. For these experiments, we define a *default* setting for all parameter of synthetic generation methods (e.g., bias parameter α , reliability parameter β , etc.). For each experiment, we fix all parameters except one; then, by varying that parameter, we aim to understand its impact on the performance of GCN-SOAN and other baselines. Our default setting includes a number of users $n = 500$ and number of items $m = 500$; random one-to-one ownership network; $\mu = (0.3, 0.7)$, $\sigma = (0.1, 0.1)$, and $\pi = (0.2, 0.8)$ for the ground-truth generation method;⁷ the number of peer grades $k = 3$,⁸ $\sigma_{max} = 0.25$, bias parameter $\alpha = 0$, and reliability parameter $\beta = 0$ for assessment network generation; and no social network generation.⁹

Figure 2a shows how the prediction error changes with the number of peer graders k while the other parameters are fixed to the default setting. Unsurprisingly, the performance

⁷This setting for ground-truth distribution is motivated by two-humped grade distribution in academic classes.

⁸This choice of 3 is motivated by the fact that most practical applications (e.g., conference review or peer grading in classrooms) do not require more than 3 peer assessment per item due to time-consuming nature of assessment processes.

⁹We have run some other experiments with different default settings. The results were qualitatively similar.

of all methods improves with k . GCN-SOAN not only outperforms others for any k , but also exhibits significant improvement over others for a relatively small k (e.g., $k \leq 4$). This superiority of GCN-SOAN with minimal number of peer graders is its strength to make peer assessment suitable and practical for different applications, as so many peer assessment requests will put unnecessary stress and burden on users, thus impeding the practicality of the system.

Figure 2b illustrates the errors for each model while changing the bias parameter α (and keeping other parameters fixed to default). GCN-SOAN performs significantly better than other models for any bias values, including generous ($\alpha > 0$) and strict graders ($\alpha < 0$). GCN-SOAN owes this success to its ability to learn students’ grading behavior by leveraging a small portion of ground truth grades and assessment network structure. These experiments show that our model could be a great choice for those peer assessment settings where the peer grades are intentionally or unintentionally overestimated/underestimated.

Figure 2c reports the errors for various values of reliability parameters β . Recall that the β controls the extent that the accuracy of each peer in his/her assessments is correlated with his/her item’s grade. Our results show that GCN-SOAN is very competitive to other models, even those built based on this correlation assumption (e.g., [27, 19]). We observe that only when $\beta > 0.8$, PeerRank outperform GCN-SOAN. One might argue that $\beta > 0.8$ is implausible scenario in practice. However, this result suggests that our model is still competitive choice for settings in which peer assessment accuracy is correlated with peer success.

To study how various ground-truth generation distribution impacts the prediction error of various method, we first change the default bimodal mixture of normal distributions (for ground truth generation) to a normal distribution by setting $\mu_1 = \mu_2$ and $\sigma_1 = \sigma_2 = 0.15$. Then, we only vary the mean of distribution while other parameters are fixed to default. As shown in Figure 2d, GCN-SOAN consistently outperforms others regardless of the underlying ground-truth distribution. Notably, PeerRank and RankwithTA do not perform well when most users own items with low grades.

Results: Synthetic Data with Strategic Assessment. We study the performance of all peer assessment methods under the strategic model discussed above. For this set of experiments, we define this default setting: number of users $n = 500$ and

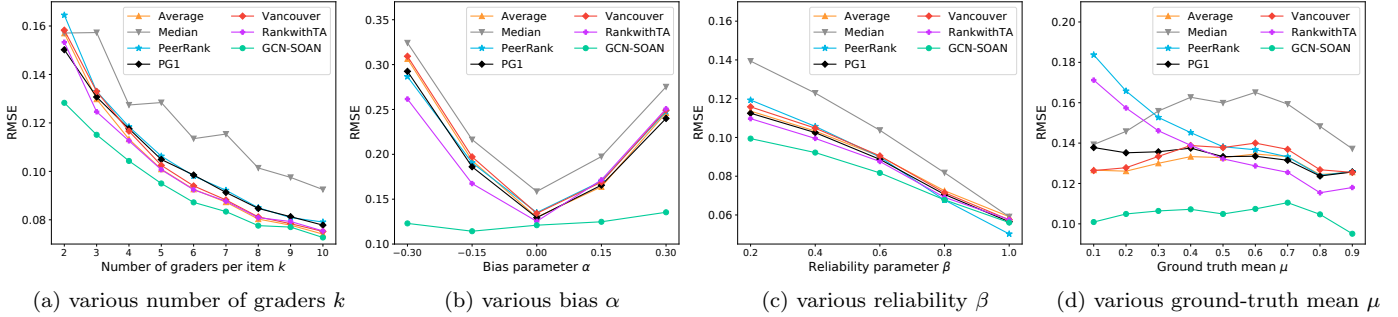


Figure 2: Root mean square errors of various methods, synthetic data with bias-reliability peer generation model, default setting for all parameters but varying (a) number of peer graders k , (b) grading bias α , (c) reliability parameter β , or (d) ground-truth mean μ . Average over four runs.

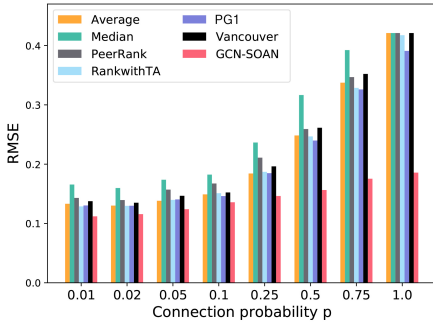


Figure 3: Average of root mean square errors, synthetic data with strategic peer grade generation model and random graph model.

number items $m = 500$; random one-to-one ownership network; $\mu = (0.3, 0.7)$, $\sigma = (0.1, 0.1)$, and $\pi = (0.2, 0.8)$ for the ground-truth generation method; the number of peer grades $k = 3$ and $\sigma_{max} = 0.25$ for assessment network generation by the strategic model; and ER random graph model with $n = 500$ and $p = 0.05$ for social network generation. We only vary the connection probability p while keeping other parameters fixed to study how the connection density of colluding social networks impact the accuracy of peer assessment methods. Figure 3 show the outstanding performance of our model compare to other benchmarks and illustrate how our model is more resilient to colluding behaviors. This result suggest that GCN-SOAN is well-equipped to detect conflict-of-interest behaviors and mitigate the possible impact of any strategic behaviors.

Discussion. Our experiments show that GCN-SOAN learns very well various grading behaviors, even when graders have intentional or unintentional biases in their evaluations. We also observe that our GCN-SOAN can outperform other benchmarks even when their main inductive biases are strongly present in the dataset (e.g., when the grading ability of users are strongly correlated to the quality of their own work).

Our set of benchmarks, in spite of being very competitive, does not cover all ML-based peer assessment methods. We make a few remarks about this. PG3 [19] is missing in our experiments as its implementation was not publicly available and we could not properly implement it to gets its competitive performance. However, we expect that GCN-SOAN outperforms PG3 since the relative improvements of GCN-

SOAN over PG1 is 10.27% (on average) compared to the average relative improvement of 1.76% for PG3 over PG1 (as reported in their original paper [19]). We also speculate that the other traditional ML-based methods might not outperform our GCN-SOAN for at least one reason: except graph neural networks, most ML methods assume that the data points (e.g., peer grades, ground-truth valuation, etc.) are identically and independently distributed (i.i.d.). However, as argued earlier, the peers’ grading behaviors, ownerships, social connections, and valuations of their owned items are all dependent on each other. Ignoring these dependencies in machine learning methods with i.i.d. assumptions make them less competitive to our GCN-SOAN in which these dependencies are well-expressed by our proposed SOAN and well-exploited by our proposed graph neural network algorithm. This might explain why the literature has even been thin in successfully exploring other advanced machine learning models and Sajjadi et al. [23] concluded that machine learning methods cannot improve over simple heuristics (e.g., average).

4. CONCLUSION AND FUTURE WORK

We represent peer assessment data as a weighted multi-relational graph, which we call social-ownership-assessment network (SOAN). Our SOAN can easily express many different peer assessment setups (e.g., self assessment, peer assessment of group or individual work, etc.). Leveraging SOAN, we introduce a modified graph convolutional network approach, which learns peer assessment behaviors, to more accurately predict ground-truth valuations. Our extensive experiments demonstrate that GCN-SOAN outperforms state-of-the-art baselines in a variety of settings, including strategic behavior, grading biases, etc.

Our SOAN model provides a solid foundation for the broader investigation of graph neural network approaches for peer assessments. Our GCN-SOAN can be extended to mitigate the over-smoothing effect observed in our experiments, or to include a different set of network weights for each relation type of social, assessment, and ownership. Another promising direction is to assess the effectiveness of GCN-SOAN or its extensions on real-world assessment data, with the presence of social network data. Finally, it would be interesting to collect data with richer node features in order to evaluate how node features can further improve our GCN-SOAN.

5. REFERENCES

- [1] H. Aziz, O. Lev, N. Mattei, J. Rosenschein, and T. Walsh. Strategyproof peer selection: Mechanisms, analyses, and experiments. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI-16)*, volume 30, 2016.
- [2] H. Aziz, O. Lev, N. Mattei, J. S. Rosenschein, and T. Walsh. Strategyproof peer selection using randomization, partitioning, and apportionment. *Artificial Intelligence*, 275:295–309, 2019.
- [3] Y. Chen, X. Tang, X. Qi, C.-G. Li, and R. Xiao. Learning graph normalization for graph neural networks. *arXiv preprint arXiv:2009.11746*, 2020.
- [4] L. de Alfaro and M. Shavlovsky. Crowdgrader: Crowdsourcing the evaluation of homework assignments. *arXiv preprint arXiv:1308.5273*, 2013.
- [5] L. De Alfaro and M. Shavlovsky. Crowdgrader: A tool for crowdsourcing the evaluation of homework assignments. In *Proceedings of the 45th ACM technical symposium on Computer science education (SIGCSE-14)*, pages 415–420, 2014.
- [6] L. De Alfaro, M. Shavlovsky, and V. Polychronopoulos. Incentives for truthful peer grading. *arXiv preprint arXiv:1604.03178*, 2016.
- [7] H. Fang, Y. Wang, Q. Jin, and J. Ma. Rankwithta: A robust and accurate peer grading mechanism for moocs. In *2017 IEEE 6th International Conference on Teaching, Assessment, and Learning for Engineering (TAL-17)*, pages 497–502, 2017.
- [8] M. Fey and J. E. Lenssen. Fast graph representation learning with pytorch geometric. *arXiv preprint arXiv:1903.02428*, 2019.
- [9] X. A. Gao, J. R. Wright, and K. Leyton-Brown. Incentivizing evaluation with peer prediction and limited access to ground truth. *Artif. Intell.*, 275:618–638, 2019.
- [10] W. L. Hamilton. Graph representation learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 14(3):1–159, 2020.
- [11] S. Jecmen, H. Zhang, R. Liu, N. B. Shah, V. Conitzer, and F. Fang. Mitigating manipulation in peer review via randomized reviewer assignments. *arXiv preprint arXiv:2006.16437*, 2020.
- [12] A. Kahng, Y. Kotturi, C. Kulkarni, D. Kurokawa, and A. D. Procaccia. Ranking wily people who rank each other. In *Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*, pages 1087–1094, 2018.
- [13] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations (ICLR-15)*, 2015.
- [14] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *Proceedings of the Fifth International Conference on Learning Representations (ICLR-17)*, 2017.
- [15] Y. Kotturi, A. Kahng, A. Procaccia, and C. Kulkarni. Hirepeer: Impartial peer-assessed hiring at scale in expert crowdsourcing markets. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI-20)*, volume 34, pages 2577–2584, 2020.
- [16] N. Miller, P. Resnick, and R. Zeckhauser. Eliciting informative feedback: The peer-prediction method. *Management Science*, 51(9):1359–1373, 2005.
- [17] A. A. Namanloo, J. Thorpe, and A. Salehi-Abari. Improving peer assessment with graph convolutional networks, 2021.
- [18] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. 2017.
- [19] C. Piech, J. Huang, Z. Chen, C. B. Do, A. Y.-T. Ng, and D. Koller. Tuned models of peer assessment in moocs. In *Proceedings of The 6th International Conference on Educational Data Mining (EDM-13)*, 2013.
- [20] C. Psenicka, W. Vendemia, and A. Kos. The impact of grade threat on the effectiveness of peer evaluations of business presentations: An empirical study. *International Journal of Management*, 30(1):168–175, 2013.
- [21] K. Reily, P. L. Finnerty, and L. Terveen. Two peers are better than one: aggregating peer reviews for computing assignments is surprisingly accurate. In *Proceedings of the ACM 2009 international conference on Supporting group work (GROUP-09)*, pages 115–124, 2009.
- [22] P. M. Sadler and E. Good. The impact of self-and peer-grading on student learning. *Educational assessment*, 11(1):1–31, 2006.
- [23] M. S. Sajjadi, M. Alamgir, and U. von Luxburg. Peer grading in a course on algorithms and data structures: Machine learning algorithms do not improve over simple baselines. In *Proceedings of the Third (2016) ACM conference on Learning@Scale (L@S-16)*, pages 369–378, 2016.
- [24] T. Staubitz, D. Petrick, M. Bauer, J. Renz, and C. Meinel. Improving the peer assessment experience on mooc platforms. In *Proceedings of the Third (2016) ACM conference on Learning@Scale (L@S-16)*, pages 389–398, 2016.
- [25] I. Stelmakh, N. B. Shah, and A. Singh. Catch me if I can: Detecting strategic behaviour in peer assessment. In *ICML Workshop on Incentives in Machine Learning*, 2020.
- [26] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. Graph attention networks. In *International Conference on Learning Representations (ICLR-18)*, 2018.
- [27] T. Walsh. The peerrank method for peer assessment. In *Proceedings of the Twenty-First European Conference on Artificial Intelligence (ECAI-14)*, page 909–914, 2014.
- [28] H. Wang and J. Leskovec. Unifying graph convolutional neural networks and label propagation. *arXiv preprint arXiv:2002.06755*, 2020.
- [29] X. Wang, M. Cheng, J. Eaton, C.-J. Hsieh, and F. Wu. Attack graph convolutional networks by adding fake nodes. *arXiv preprint arXiv:1810.10751*, 2018.
- [30] Y. Wang, H. Fang, Q. Jin, and J. Ma. Sspa: an effective semi-supervised peer assessment method for large scale moocs. *Interactive Learning Environments*, pages 1–19, 2019.
- [31] Y. Wang, Z. Hu, Y. Ye, and Y. Sun. Demystifying graph neural network via graph filter assessment. 2019.

- [32] J. R. Wright, C. Thornton, and K. Leyton-Brown. Mechanical ta: Partially automated high-stakes peer grading. In *Proceedings of the 46th ACM Technical Symposium on Computer Science Education (SIGCSE-15)*, pages 96–101, 2015.
- [33] Q.-S. Xu and Y.-Z. Liang. Monte carlo cross validation. *Chemometrics and Intelligent Laboratory Systems*, 56(1):1–11, 2001.
- [34] Y. Xu, H. Zhao, X. Shi, and N. B. Shah. On strategyproof conference peer review. In *IJCAI*, pages 616–622. ijcai.org, 2019.
- [35] H. Zarkoob, H. Fu, and K. Leyton-Brown. Report-sensitive spot-checking in peer-grading systems. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS-20)*, pages 1593–1601, 2020.

Using Neural Network-Based Knowledge Tracing for a Learning System with Unreliable Skill Tags

Shamya Karumbaiah
Carnegie Mellon University
shamya@cmu.edu

Jiayi Zhang
University of Pennsylvania
jzhang7718@gmail.com

Ryan S. Baker
University of Pennsylvania
ryanshaunbaker@gmail.com

Richard Scruggs
Karolinska Institute
richardtscruggs@gmail.com

Whitney Cade
American Institutes for Research
wcade@air.org

Margaret Clements
American Institutes for Research
pclements@air.org

Shuqiong Lin
American Institutes for Research
slin@air.org

ABSTRACT

Considerable amount of research in educational data mining has focused on developing efficient algorithms for Knowledge Tracing (KT). However, in practice, many real-world learning systems used at scale struggle to implement KT capabilities, especially if they weren't originally designed for it. One key challenge is to accurately label existing items with skills, which often turns out to be a herculean task. In this paper, we investigate whether an increasingly popular approach to knowledge tracing, the use of neural network models, can be a partial solution to this problem. We conducted a case study within a commercial math blended learning system. Using the data collected from middle school students' use of the system over two years, we compare the performance of a neural network-based KT model (DKVMN) in three scenarios: 1) with the original (possibly unreliable) system-provided skill tags, 2) with coarser-grained domain tags based on state standards, and 3) without inputting any mappings between content and skills. Our results suggest that including the system-provided skills in the training of the model leads to the worst performance. The best performance is observed when the skills are entirely disregarded. This supports the possibility of bypassing the laborious step of item-skill tagging in real-world learning systems which were not originally designed to work with KT models, especially if the goal is only to predict the performance of a student on future items. We discuss the implications of our findings for practice and future research.

Keywords

Knowledge Tracing, Skill Tagging, Deep Learning, DKVMN, Adaptive Learning

1. INTRODUCTION

Knowledge tracing has become an essential part of modern adaptive learning systems and even many non-adaptive learning

systems. The ability to infer a students' latent knowledge of a skill -- or at least predict their performance on future items within the skill -- has several applications. One of the key applications of knowledge tracing is mastery learning [7]. A system using knowledge tracing for mastery learning divides content into skills (sometimes also called concepts or knowledge components), and once a student starts a skill they cannot advance beyond it without demonstrating that they have mastered the skill. Knowledge tracing is also often used for creating reports for teachers [31] or open learner models for students [4]. A third use of knowledge tracing is as a component in automated detectors that recognize a range of student behaviors or states, including help-seeking strategies [1], and disengaged behaviors such as gaming the system [18].

While the first learning systems to use knowledge tracing were designed with that function in mind [2], many real-world learning systems used at scale today were not designed with knowledge tracing or adaptive learning originally in mind. Increasingly, these systems are being retrofit to use some form of these types of functionality [16]. One of the key ways that these systems often differ from learning systems originally designed for use with knowledge tracing is in how content is mapped to skills (often referred to as skill-item mappings or as KC mappings -- KC stands for knowledge component -- [13]). Systems designed from the start to use KT first select what skills are to be included, and then develop items tailored to those skills [2]. Afterwards, there may even be a process of using data to refine the KC mapping, attempting to improve the correlation of items to each other while maintaining human comprehensibility of the overall mapping [12, 14, 15]. By contrast, when content is retrofit for use with knowledge tracing, items are created first, and then the items are labeled with skills.

Labeling an existing item with a skill is much harder than creating a new item for a skill. Often, items have been developed by multiple authors over time, or have come from different original sources such as different textbooks. Mapping this disparate content -- sometimes tens of thousands of items -- to a set of skills can be a herculean task. In many cases, items have been tagged in terms of governmental curricular standards, but these standards are typically much coarser-grained than the types of skills used within knowledge tracing models [6]. Therefore, there is a challenge to using many of the classic approaches to knowledge tracing, used at

S. Karumbaiah, J. Zhang, R. Baker, R. Scruggs, W. Cade, M. Clements, and S. Lin. Using neural network-based knowledge tracing for a learning system with unreliable skill tags. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 333–338, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852994>

scale in past real-world systems, with content not initially designed with this use in mind.

In this paper, we investigate whether an increasingly popular approach to knowledge tracing, the use of neural network models [12, 17, 33], can be a partial solution to this problem. Unlike earlier approaches to knowledge tracing (i.e. [8, 19]), neural network models do not require a KC model in order to predict student performance on future items. As such, it may be possible to bypass the step of developing a KC model entirely, at least for goals such as choosing what problem to give the student next. In investigating this, we select the DKVMN (Dynamic Key-Value Memory Network – [33]) algorithm, a generally successful early approach to neural network-based knowledge tracing, as it has the ability to both use an existing KC model *and* to fit a new KC model, giving us the ability to compare between using a KC model known to have considerable limitations and using no KC model at all.

Towards this goal, we conduct a case study within a commercial math blended learning system. Using the data collected from middle school students’ use of the system over two years, we compare the performance of KT models in three scenarios: 1) with the original (possibly unreliable) system-provided skill tags, 2) with coarser-grained domain tags based on state standards, and 3) without inputting any mappings between content and skills. Our results suggest that including the system-provided skills in the training of the model leads to the worst performance. The best performance is observed when the skills are entirely disregarded. This supports the possibility of bypassing KC modelling in real-world learning systems which were not originally designed to work with KT models, especially if the goal is only to predict the performance of a student on future items.

2. MOTIVATION

2.1 Knowledge Tracing

There have been a range of algorithms used for knowledge tracing over the last three decades. The first widely-used knowledge tracing algorithms relied on a KC model, as mentioned in the introduction. Perhaps the first widely-used algorithm for knowledge tracing was Bayesian Knowledge Tracing [8], based on a simple Markov Model (and also mathematically equivalent to a simple Bayesian Network – [27]). More recently, models based on logistic regression have become popular in the literature [5, 20] although they remain rare within real-world use (but see [16]). Algorithms related to item response theory (IRT) such as Elo [21] and temporal IRT [30] have also become more widely seen in the literature recently, and are used at scale in several learning systems [3, 22, 30]. While Elo and temporal IRT can be used without a KC model, typically a separate Elo model is used for each of several skills.

A contrasting set of approaches uses neural networks to avoid the need for a KC model. The first member of this family of algorithms, deep knowledge tracing [24], discovered complex item-to-item mappings as part of predicting future performance on items. Later approaches such as DKVMN introduced the ability to use or fit skill-item mappings (KC models) as well [33]. The last five years have seen a proliferation of knowledge models based on neural networks [11, 17, 29, 33], gaining increasing ability to predict future performance. However, due to concerns about unpredictable behavior [9, 32], and the challenge of using this type of models for mastery learning and reporting on student skill (as discussed above, the main applications of knowledge tracing in contemporary learning systems), neural networks have been much more popular in the published literature on knowledge tracing than in actual real-world

use. One of the key limitations to neural network models in this regard has been that they predict correctness on specific problems but do not map that back to inferring proficiency on human-interpretable skills. Recent extensions have taken this step, suggesting the potential for broader real-world use and uptake of neural network knowledge tracing models. In particular, an extension in [28] can be applied to any neural network knowledge tracing model for which there is a KC model available (even at the coarse level of state standards).

2.2 Case Study: A Commercial Learning System with Imperfect Skill Tags

This study uses data from a commercial math blended learning platform used in schools across the United States by students from kindergarten through Algebra II. This platform allows teachers to assign problem sets to students by standards and topics, and assignments can be personalized based on the teacher’s assessment of a student’s progress. The system measures a student’s mastery of underlying mathematical concepts through Bayesian Knowledge Tracing (BKT), and may suggest more foundational or advanced assignments based on a student’s performance. In this way, the system contains several elements frequently seen in intelligent tutoring systems. Assignments may be worked on at home or at school, and guidance provided by the creators of the platform suggest that students should work in the platform for approximately 30 minutes per week.

The learning platform has two classes of problems, authored in different ways. In the “classic” problems, students answer multiple-choice, multi-step word problems, with a series of scaffolded, smaller word problems triggered when the student answers the initial problem incorrectly. The second “new” class of problems tends to be more interactive: for example, students plot points and lines on a graph, rotate shapes, and sort statements or numbers using drag-and-drop functionality. These problems tend to be open-response and typically lack the kind of scaffolding seen in the classic problems.

The difference in the content within this system can be attributed to the development of the content in two phases. In a first phase of development, a team of math content specialists created multiple-choice items with scaffolding for incorrect answers (based on the model in [26]). In this model, students that answer an item correctly will move onto the next problem, but if a problem is answered incorrectly initially, the problem will be broken into a series of additional, smaller questions that scaffold an effective set of problem-solving steps. The math content specialists tagged each item with a set of skills based on mathematics standards. After the acquisition of the system by a different company, and the subsequent departure of the original development team, a different approach was adopted to build out additional content for the same courses. In this new approach, existing content created by an outside company was added to the system. This additional content had more item types, as discussed above. However, this new content did not use the same scaffolding model as the original content. Items had been tagged by the outside company according to the same overall mathematics standards, but there were considerable inconsistencies and incompatibilities between the two tagging approaches. The result was an overall set of skill tags that were of uncertain usability.

While the case of a system being acquired by a different company and adopting a new process for content is perhaps somewhat unusual, the overall problem of content and skill tags being developed by different teams over time is not unusual. Many contemporary online learning systems integrate data from different textbooks,

shift the membership of content authoring teams over the span of many years, and tag content according to multiple state standards and internal content schemas. While many of the systems which are most heavily published on at this conference (and related conferences) do not have these issues, it is likely that the majority of learning systems used at large scale suffer from these issues to at least some degree.

3. METHODS

3.1 Data Collection

The analyses presented here are based on data collected as part of a larger, exploratory study on the implementation of the blended learning platform discussed in section 2.2. Though the focus of the study was on 6th-8th grade students, we also collected data from sections of Algebra I and Geometry students when a teacher taught across several grade levels and wanted to use the platform across their classes. These data were collected in the 2018-19 and 2019-20 school years from students located in 19 schools across Texas. These schools tended to be located in less populated areas, with 11% of schools in the sample in cities (according to Federal classification), 5% of schools in the sample in suburbs, 26% in towns, and 58% in rural areas. 41% of the students in our sample schools identified as white, 9% as African-American/Black, 46% as Hispanic, and 1% as Asian. 60% of students in sample schools qualified for free/reduced price lunches.

Data collected occurred across two years, with 38 teachers and 2069 students participating in the first year, and 14 teachers and 743 students participating in the second year. Teachers were instructed to have students engage with the blended learning software for approximately 30 minutes each week, though use of the system diminished over the course of the school year (cf. [23]). The primary goal of the larger study was to examine the naturalistic relationships between learning, engagement, and classroom implementation, so researchers interfered very little in how teachers and students decided to use the platform. Teachers used the software differently: some teachers sat their students in front of the system, offering little help, while others used the system while teaching to the class, talking about the problems presented. Some teachers disliked the scaffolding problems and checked students' answers to top-level problems to avoid their students receiving scaffolding problems. In some classes, students asked each other for help, while in others they were expected to work alone.

3.2 Descriptive Statistics

This analysis involves data from 2564 students. Table 1 presents the number of problems, skills, and first attempts for classic and new content. The overall correctness of students' first attempts on problems is 32% across all the problems. However, when the content is categorized based on the new and classic content (see description in Section 2.2), we see that the percentage correctness is higher for classic content (59%) with the new content having a much lower correctness (15%). There are several possible explanations for this finding. Students may be more likely to get scaffolding problems correct than initial problems, as scaffolding problems are designed to have lower difficulty. In addition, the more interactive problems often require the student to take several actions or input multiple steps in order to arrive at the correct final answer, which creates more opportunities for mistakes. Due to the clear differences in these two types of learning content, we will look at these separately in our final experiment.

Table 1. Total number of problems, skills, first attempts, and percentage correctness for all content, classic content, and new content in the data used in this analysis

Content Type	#problems	#skills	#first attempts	%correct
All content	7252	2,237	110,214	32%
Classic content only	1764	532	42,353	59%
New content only	5488	1,705	67,861	15%

3.3 DKVMN

In this analysis, we use Dynamic Key-Value Memory Networks (DKVMN; [33]) for knowledge tracing. DKVMN is a KT model developed based on neural networks and has demonstrated improved performance compared to traditional KT models, such as Bayesian Knowledge Tracing or Performance Factors Analysis [25, 28, 33]. In DKVMN, the model employs two matrices to predict student performance on items and estimate mastery on a set of automatically-derived skills. By learning from the relationships between the two matrices (i.e. a static matrix that stores skill relationships and a dynamic matrix that stores and updates mastery level), the algorithm generates underlying skills associated with items and identifies connections between them, creating a skill map. DKVMN utilizes the map to make predictions on student performance and estimates mastery learning, as opposed to the human-generated skill-item mapping that the traditional KT models rely on. Given DKVMN's capability of automatically generating underlying skills and exploiting the relationships between them, we are interested in investigating how the algorithm can be applied to handle data with unreliable skill tagging.

4. ANALYSIS

The goal of our analysis in this paper is to investigate the use of a neural network-based KT model (DKVMN) that doesn't require skill tags, for a system where the skill tags may be unreliable or inconsistent. Accordingly, we compare the performance of KT models with and without mappings between content and skills. Since we don't have out-of-system performance data on students to validate knowledge estimates, we infer performance within the system rather than outside the system (note that DKVMN can also be used to predict out-of-system performance – [28]). Thus, we validate the model based on its ability to predict a student's correctness on the next problem. We use [33]'s implementation of the DKVMN model with a set of hyperparameters that have been reported to produce optimal outcome for a previously collected dataset (the ASSISTments2009 data set in [33]), including a state dimension of 50 and a memory size of 20. We use this previously established set of hyperparameters instead of tuning them, to avoid overfitting. In addition, since the goal of the paper is to study whether the skill tags are useful and whether DKVMN can compensate for a lack of good skill tags in a data set of this nature, tuning the hyperparameters to find the best performing KT model is not related to the goals of the analysis. We evaluate our model using AUC ROC. We perform a student-level 10-fold cross validation, and the reported results are the average across the CV folds.

Using this approach, we conduct the following three kinds of analysis:

- 1) Experiment #1 (Default-Skills): In this experiment, we chose to include the initial skill mapping provided by the

learning system during the training of the DKVMN model. The system provides a total of 2237 skills.

- 2) Experiment #2 (**Domains**): In this experiment, instead of using the default skills, we group the skill tags provided by the learning system into coarser-grained domains defined by the Common Core Standards [6]. We derive domain information from the nomenclature used within the standards (e.g., exponential functions, quadratic equations, and two-dimensional shapes). This reduced the 2237 skills to 24 domains.
- 3) Experiment #3 (**No-Skill**): In this experiment, we disregard the skill tags entirely and instead treat all problems as a single skill. As discussed above, DKVMN will then find a latent mapping of skills to the problems on its own.

We repeat these three experiments for three data setups: 1) with all content, 2) classic content only, and 3) new content only. In the last two settings, we separate the training and testing data to only include new content or classic content since we have some evidence that data for these two types of content is qualitatively different (see discussion in Section 2.2).

5. RESULTS

Table 2 presents the results of our experiments. Here we summarize our observations:

- 1) Experiment #1 (**Default-Skills**): We observe that including the system-provided skills in the training of the DKVMN model leads to the worst performance in all three cases (all content, classic content, new content). For all content and classic content models, the model performance as measured by cross-validated AUC ROC is between 0.5 and 0.6 (0.564 and 0.532), relatively modest improvements on chance and much lower than is typically seen for DKVMN (e.g. [25, 33]).
- 2) Experiment #2 (**Domains**): When the system-provided skills are grouped into coarser-grained domain mappings, we observe a big improvement in DKVMN’s performance. For example, there is a 0.206 increase in AUC when the model trained on all content uses the domain tags instead of skill tags (0.770 vs. 0.564). Similarly, there is a big increase in AUC of 0.233 for the model trained on new content (0.900 vs. 667). However, there is a relatively smaller increase in AUC of 0.084 for the model trained on classic content (0.616 vs. 0.532).
- 3) Experiment #3 (**No-Skill**): Finally, disregarding the skills entirely leads to the best performance in all three cases: 0.821 for all content, a still relatively unimpressive 0.634 for classic content, and 0.920 for new content.
- 4) If we compare the results of the DKVMN models separately for the new and classic content, we observe that the model trained only on new content perform much better than models trained only on classic content (e.g. 0.667 vs. 0.532 with default skills and 0.920 vs. 0.634 with no skills)

Table 2. Average AUC of DKVMN models across 10-fold cross-validation for the four experiments conducted using default skills (#1), domains (#2), no skills (#3), and for new and classic content separately.

Content Type	Default-Skills (exp #1)	Domains (exp #2)	No-Skill (exp #3)
All content	0.564	0.770	0.821
Classic content only	0.532	0.616	0.634
New content only	0.667	0.900	0.920

Overall, we also see that the DKVMN model trained without the skill information on all content (0.821) or new content (0.920) has a comparable or better performance than the best-performing models reported on some benchmark datasets (0.827 for Synthetic-5, 0.816 for ASSISTments2009, 0.727 for ASSISTments2015, 0.828 for Statics2011; [33]). With the exception of ASSISTments2015 dataset, the best performing models for the other three datasets have achieved AUCs between 0.80 and 0.83. Our best-performing model for all content combined (0.821) is comparable to the performance of the best models on these benchmark datasets. The best model trained only on new content has higher performance (0.920) than the best models on benchmark datasets. However, the best model for the classic content has much lower performance (0.634) than previous uses of DKVMN.

6. DISCUSSION

Knowledge tracing models are often used in learning systems to estimate students’ knowledge of a skill. In some cases, this is operationally defined as simply predicting whether or not a student will get the next problem (or a specific problem) right. Accordingly, considerable amount of research has focused on developing efficient algorithms for knowledge tracing. However, in practice, many real-world learning systems used at scale are difficult to implement knowledge tracing for, especially if the system or content was not originally designed for use with a skill model. The key challenge is to accurately label existing items with skills, which often turns out to be a herculean task (Section 1). Little research has explored ways to address this practical constraint limiting the use of knowledge tracing models at scale. In this paper, we investigate whether an increasingly popular approach to knowledge tracing, the use of neural network models, can be a partial solution to this problem.

Our analysis investigates the use of a neural network-based KT model (DKVMN) that doesn’t require skill tags (and can even automatically assign its own skill tags) to bypass the step of retrofitting content with skills. We conduct a case study of a commercial math blended learning system which has a potentially unreliable and/or inconsistent skill tagging, due to the content and skill tags being developed by different teams over time (Section 2.2). We collected data from 6th-8th grade students’ system usage over two years within 19 schools across Texas. We compare the performance of DKVMN in three scenarios: 1) with the potentially unreliable default system-provided skill tags, 2) with coarser-grained domain tags based on state standards, and 3) without inputting any mappings between content and skills. We also investigate differences in predictive performance for two disparate content types in the system: classic content (multiple choice, with scaffolds) and new content (more interactive, open-response, with no scaffolds).

Our results suggest that including the system-provided default skills in the training of the DKVMN model leads to the worst performance at predicting future student performance within the learning system. The AUC ROC for this case is much lower than what is typically seen for DKVMN. Big improvements in performance are observed when the system-provided skills are grouped

into coarser-grained domain mappings. However, the best performance is obtained when the skills are entirely disregarded. To our surprise, there is a noticeable difference in the improvement between the classic and new content, with relatively smaller increase in AUC for the model trained on classic content.

6.1 Implications

The objective of our investigations was to explore the possibility of bypassing the herculean task of item-skill tagging (KC model) for real-world learning systems which were not originally designed to work with KT models. Our results provide some support to this possibility, especially if the goal is only to predict the performance of a student on future items. For example, if a KT model is being developed for a system for optimal next problem selection, using a neural network-based model like DKVMN and ignoring the skill tags may be more effective than using unreliable system-provided skills.

However, our results were not consistent across all content, suggesting that this approach may not be sufficient for all legacy content. Despite using DKVMN, the best model for classic content still achieves an AUC that is much lower than is typically seen for neural network-based KT models. Therefore, careful consideration is needed before making the decision on deploying such a model to make real-time decisions on what content the student will see next. In comparison, the DKVMN model trained on new content without any skill information is at par with the best models on benchmark KT datasets. It is difficult at this point to explain why DKVMN was so much more successful for new content than for classic content. There is no obvious reason why either of the two clear attributes of the older content – scaffolding problems and the use of multiple-choice – would lead to poorer prediction by DKVMN, given the past successful use of DKVMN and related algorithms on content with these attributes.

Though these findings suggest that DKVMN may be useful for legacy systems with unreliable or inconsistent skill tags, it is important to keep a few things in mind before assuming that these findings will apply to other contexts. First, this is a case study of one commercial learning system. These results need to be replicated in other content types, system design, subject matters, student demographics, etc. We have made the code public at (redacted for review) to aid replication. Second, a better performing model may still not necessarily serve all student subgroups equitably. Before actually applying an algorithm, it is necessary to investigate it for potential biases that could lead to discriminatory behaviors. In this case, a biased KT model could deliver content below a student's actual skill level more often for students from certain subgroups, leading to missed learning opportunities. Since neural network models are prone to overfitting due to their complex decision boundaries, it is particularly important to investigate whether a model like DKVMN generalizes less well for students who are unrepresented in the training data. For instance, it may be that a neural network model captures complex interrelationships between skills that only occur in students with specific past curricular experiences.

6.2 Limitations and Future Work

There are a few limitations to the analysis conducted in this paper. First, since the learning system itself made pedagogical decisions based on a knowledge tracing model using unreliable skill tags, the problems it gave students were probably not always appropriately chosen. As such, the data may not represent students participating in mastery learning (although this is also true of the data sets used to initially train KT for mastery learning in other commercial systems). Second, we chose to explore DKVMN because it has

generally been reported to be successful, and generates its own skill mapping. A more comprehensive comparison between other neural network-based models could be helpful in understanding whether other algorithms can perform better for the old content. Lastly, before disregarding non-neural network-based KT models for the case discussed in this paper, it may be worth comparing these results with newer extensions proposed for classic KT models. For example, future work could explore BKT variants that include skill refitting (cf. [12]).

Future work also may be able to shed more light on the contexts where this approach does and does not work. For instance, why is there a noticeable difference in the improvement between the classic and new content with relatively smaller increase in AUC for the model trained on classic content? Exploring answers to this question may help identify cases where this approach may be more efficient than others and identify ways to improve it.

Finally, like most neural network models, DKVMN lacks easy interpretation. This could make it harder for instructors to trust its recommendations and hard to troubleshoot. A potential solution is to interpret the output of the DKVMN model. Considering that DKVMN creates its own internal tag-concept mapping, this study's results suggest that it may be worth studying those mappings in greater detail. Since the no-skill model outperformed the default-skill model to such a degree in this dataset, it is likely that DKVMN's distilled concepts represent accurate relationships between the problems in the data. Zhang and colleagues [33] mention the possibility of using DKVMN for concept discovery, but most subsequent work on the algorithm has instead focused on its predictive accuracy (but see [10], who discuss interpretability). Additional research on DKVMN's process of concept discovery could not only improve its interpretability but potentially also allow for better automated skill tagging and more accurate student skill level estimation.

7. ACKNOWLEDGMENTS

The research reported here was supported by the IES grant R305A170167. The opinions expressed are those of the authors and do not represent views of IES.

8. REFERENCES

- [1] Aleven, V., McLaren, B., Roll, I., & Koedinger, K. (2006). Toward meta-cognitive tutoring: A model of help seeking with a Cognitive Tutor. *International Journal of Artificial Intelligence in Education*, 16(2), 101-128.
- [2] Anderson, J. R., Corbett, A. T., Koedinger, K. R., & Pelletier, R. (1995). Cognitive tutors: Lessons learned. *The journal of the learning sciences*, 4(2), 167-207.
- [3] Brinkhuis, M., Cordes, W., & Hofman, A. (2020). Governing games Adaptive game selection in the Math Garden. In *ITM Web of Conferences* (Vol. 33, p. 03003).
- [4] Bull, S., & Mabbott, A. (2006, June). 20000 inspections of a domain-independent open learner model with individual and comparison views. In *International conference on intelligent tutoring systems* (pp. 422-432). Springer, Berlin, Heidelberg.
- [5] Choffin, B., Popineau, F., Bourda, Y., & Vie, J. J. (2019, July). DAS3H: Modeling Student Learning and Forgetting for Optimally Scheduling Distributed Practice of Skills. In *International Conference on Educational Data Mining (EDM 2019)*.
- [6] Common Core (2022) Mathematics Standards. Accessed March 8, 2022 from <http://www.corestandards.org/Math/>.

- [7] Corbett, A. (2001). Cognitive computer tutors: Solving the two-sigma problem. In *International Conference on User Modeling* (pp. 137-147).
- [8] Corbett, A. T., & Anderson, J. R. (1995). Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4(4), 253-278.
- [9] Ding, X., & Larson, E. C. (2019). Why Deep Knowledge Tracing Has Less Depth than Anticipated. *Proc. International Conference on Educational Data Mining*.
- [10] Ding, X., & Larson, E. C. (2021). On the Interpretability of Deep Learning Based Models for Knowledge Tracing. ArXiv:2101.11335 [Cs].
- [11] Ghosh, A., Heffernan, N., & Lan, A. S. (2020, August). Context-aware attentive knowledge tracing. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining* (pp. 2330-2339).
- [12] Khajah, M., Lindsey, R. V., & Mozer, M. C. (2016). How Deep is Knowledge Tracing?. *Proc. International Conf Educational Data Mining*.
- [13] Koedinger, K. R., Corbett, A. T., & Perfetti, C. (2012). The Knowledge-Learning-Instruction framework: Bridging the science-practice chasm to enhance robust student learning. *Cognitive science*, 36(5), 757-798.
- [14] Koedinger, K. R., Stamper, J. C., McLaughlin, E. A., & Nixon, T. (2013, July). Using data-driven discovery of better student models to improve student learning. In *International conference on artificial intelligence in education* (pp. 421-430).
- [15] Liu, R., & Koedinger, K. R. (2017). Closing the Loop: Automated Data-Driven Cognitive Model Discoveries Lead to Improved Instruction and Learning Gains. *Journal of Educational Data Mining*, 9(1), 25-41.
- [16] Maier, C., Baker, R.S., Stalzer, S. (2021) Challenges to Applying Performance Factor Analysis to Existing Learning Systems. *Proceedings of the 29th International Conference on Computers in Education*.
- [17] Pandey, S., & Karypis, G. (2019). A self-attentive model for knowledge tracing. *arXiv preprint arXiv:1907.06837*.
- [18] Paquette, L., Baker, R.S. (2019) Comparing machine learning to knowledge engineering for student behavior modelling: A case study in gaming the system. *Interactive Learning Environments*, 585-597.
- [19] Pavlik, P. I., Cen, H., & Koedinger, K. R. (2009). Performance Factors Analysis--A New Alternative to Knowledge Tracing. In *Proceedings of the 2009 conference on Artificial Intelligence in Education*.
- [20] Pavlik, P. I., Eglington, L. G., & Harrell-Williams, L. M. (2021). Logistic Knowledge Tracing: A Constrained Framework for Learner Modeling. *IEEE Transactions on Learning Technologies*, 14(5), 624-639.
- [21] Pelánek, R. (2016). Applications of the Elo rating system in adaptive educational systems. *Computers & Education*, 98, 169-179.
- [22] Pelánek, R., Papoušek, J., Řihák, J., Stanislav, V., & Nižnan, J. (2017). Elo-based learner modeling for the adaptive practice of facts. *User Modeling and User-Adapted Interaction*, 27(1), 89-118.
- [23] Phillips, A., Pane, J. F., Reumann-Moore, R., & Shenbanjo, O. (2020). Implementing an adaptive intelligent tutoring system as an instructional supplement. *Educational Technology Research and Development*, 68(3), 1409-1437.
- [24] Piech, C., Bassen, J., Huang, J., Ganguli, S., Sahami, M., Guibas, L. J., & Sohl-Dickstein, J. (2015). Deep knowledge tracing. *Advances in neural information processing systems*, 28.
- [25] Pu, S., Converse, G., & Huang, Y. (2021, June). Deep Performance Factors Analysis for Knowledge Tracing. In *International Conference on Artificial Intelligence in Education* (pp. 331-341).
- [26] Razzaq, L., & Heffernan, N. T. (2006). Scaffolding vs. hints in the Assistment System. In *International Conference on Intelligent Tutoring Systems* (pp. 635-644).
- [27] Reye, J. (2004). Student modelling based on belief networks. *International Journal of Artificial Intelligence in Education*, 14, 63-96.
- [28] Scruggs, R., Baker, R.S., McLaren, B.M. (2020) Extending Deep Knowledge Tracing: Inferring Interpretable Knowledge and Predicting Post System Performance. *Proceedings of the 28th International Conference on Computers in Education*.
- [29] Shin, D., Shim, Y., Yu, H., Lee, S., Kim, B., & Choi, Y. (2021, April). Saint+: Integrating temporal features for ednet correctness prediction. In *LAK21: 11th International Learning Analytics and Knowledge Conference* (pp. 490-496).
- [30] Wilson, K. H., Karklin, Y., Han, B., & Ekanadham, C. (2016). Back to the Basics: Bayesian Extensions of IRT Outperform Neural Networks for Proficiency Estimation. *Proc. International Conference on Educational Data Mining*.
- [31] Khakaj, F., Alevan, V., & McLaren, B. M. (2017, September). Effects of a teacher dashboard for an intelligent tutoring system on teacher knowledge, lesson planning, lessons and student learning. In *European conference on technology enhanced learning* (pp. 315-329).
- [32] Yeung, C. K., & Yeung, D. Y. (2018, June). Addressing two problems in deep knowledge tracing via prediction-consistent regularization. In *Proceedings of the Fifth Annual ACM Conference on Learning at Scale* (pp. 1-10).
- [33] Zhang, J., Shi, X., King, I., & Yeung, D. Y. (2017, April). Dynamic key-value memory networks for knowledge tracing. In *Proceedings of the 26th international conference on World Wide Web* (pp. 765-774).

#lets-discuss: Analyzing Student Affect in Course Forums Using Emoji

Ariel Blobstein
Ben-Gurion University
arielblo@post.bgu.ac.il

Kobi Gal
Ben-Gurion University
University of Edinburgh
kobig@bgu.ac.il

Hyunsoo Gloria Kim
University of California, Davis
hyun@ucdavis.edu

Marc Facciotti
University of California, Davis
mtfacciott@ucdavis.edu

David Karger
MIT
karger@mit.edu

Kamali Sripathi
University of California, Davis
ksripathi@ucdavis.edu

ABSTRACT

Emoji are commonly used in social media to convey attitudes and emotions. While popular, their use in educational contexts has been sparsely studied. This paper reports on the students' use of emoji in an online course forum in which students annotate and discuss course material in the margins of the online textbook. For this study, instructors created 11 custom emoji-hashtag pairs that enabled students to quickly communicate affects and reactions in the forum that they experienced while interacting with the course material. Example reporting includes, inviting discussion about a topic, declaring a topic as interesting, or requesting assistance about a topic. We analyze emoji usage by over 1,800 students enrolled in multiple offerings of the same course across multiple academic terms. The data show that some emoji frequently appear together in posts associated with the same paragraphs, suggesting that students use the emoji in this way to communicating complex affective states. We explore the use of computational models for predicting emoji at the post level, even when posts are lacking emoji. This capability can allow instructors to infer information about students' affective states during their "at home" interactions with course readings. Finally, we show that partitioning the emoji into distinct groups, rather than trying to predict individual emoji, can be both of pedagogical value to instructors and improve the predictive performance of our approach using the BERT language model. Our procedure can be generalized to other courses and for the benefit of other instructors.

Keywords

Affect recognition, Course Forums

1. INTRODUCTION

Students typically experience a range of affective responses to learning materials, taking the form of emotions, beliefs

A. Blobstein, K. Gal, D. Karger, M. Facciotti, H. Kim, J. Almahmoud, and K. Sripathi. #lets-discuss: Analyzing student affect in course forums using emoji. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 339–345, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853101>

and attitudes that have a profound effect on their learning (and vice versa) [7]. Understanding what students are feeling when they learn can, therefore, be useful for teachers who may wish to adapt course content and/or delivery in response to student affect [26, 18, 1].

Emoji are conventionally used in social media to enhance the meaning of text or to be used as substitute for words [17]. When conversing through text, such as with written posts in course forums, emoji can be a natural way for students to express what they are feeling about the learning material. A body of work in the social and computational sciences has studied people's use of emoji, and used them to facilitate computational tasks such as sentiment analysis and emotion recognition [19, 13]. However, the use of emoji by students in educational forums has been much less studied. Our contribution is a computational analysis of students' use of emoji in a course forum with over 1,800 of students.

Students in this course use the Nota Bene (NB) collaborative annotation-based forum, which allows them to anchor messages directly to reading material in their online course textbook. Course instructors designed a set of 11 emoji that allow students to communicate affective responses in their posts, such as expressing curiosity or confusion about topics in the course, or inviting discussion about a topic. Students were incentivized to use these emoji by receiving course points on their reading assignment for using at least one emoji in their forum posts.

We provide a detailed analysis of emoji usage in the course at the post level as well as the paragraph level in the reading material that the post refers to. We find that the most commonly used emoji request discussion and assistance from instructors or peers, and that in some cases students combine several emoji in the same post to create complex affects. Some emoji frequently appear together in posts that are anchored in the same paragraph, suggesting that students may simultaneously experience a combination of responses to the course readings or that they choose to use multiple emoji to express more complex affects that are not sufficiently well represented by a single emoji.

We explore the use of computational models for classifying emoji use at the individual and group level. We use a

pre-trained deep-learning language model (BERT) to predict emoji at the post level. This model significantly outperforms a Long-Short-Term-Memory (LSTM) architecture that was used by others to predict emoji in social media model, but was trained on the same task. [13, 3].

In agreement with course instructors, we cluster the emoji into categories, based on the relationship between the predicted emoji probabilities that are outputted by the BERT model. We partition the emoji into eight distinct groups (3 groups containing an emoji pair, and 5 groups containing a single emoji). Partitioning the emoji into these distinct groups makes pedagogical sense to the course instructors while also improving performance when predicting emoji-groups rather than individual emoji.

The contribution of this work is in a new computational model for predicting student affect that was trained on students' self-reported emoji without requiring hand-labeling of the data by experts. The model can directly be applied to any discussion forum, even where emoji information is not readily available, potentially increasing its impact. Our work can help instructors, particularly in high-enrollment courses, decide where and how to intervene in discussions, and to assess where content or course revisions might be made to improve student learning.

2. RELATED WORK

Our work relates to past studies for inferring student affect from their online text conversations, as well as computational work that analyzes the use of emoji in social media. We mention relevant prior works below and refer the reader to the review of the field by Kastrati et al. [16] for additional details.

Basic approaches to infer students' sentiments in online conversations used classic NLP methods (parsing, lexical dictionaries) [8, 6]. Studies showing the correlation between affects and dropout [28] and the correlation between affects and exams scores exist [23]. Jena [15] used classical machine learning methods (SVMs, Naive Bayes) to learn the sentiment polarity (positive, negative, and neutral) from students' posts, as well as predicting basic students' emotions from text (e.g., anxiety, bored, confused, excited). Estrada et al. [12] used deep neural networks, such as convolutional neural network and LSTMs, as well as evolutionary generative models, in order to classify sentiment polarity and emotions.

As noted by Kastrati et al. there are relatively few works on recognizing students' emotions from text, despite the pedagogical importance of this task, and the growing prevalence of online learning [26, 18]. One possible reason for this scarcity of works is the reliance on hand-labeled data sets which are costly and time-consuming to obtain. We directly address this gap in our work by using students' self-reported affect in the form of emoji as proxies for their emotional state.

There is growing research studying the use of emoji as tools in computer-mediated communication [25, 17]. Emoji can be used in two different ways. They can appear alongside text to include an affect or to enhance the meaning of a

text. They can also be used as a substitute for words, or to conform a new notion.

Zhang et al. [29] investigated the use of emoji among students in the Nota Bene framework and showed the potential of detecting students' affects by training a classifier in order to distinguish between confusion and curiosity. Geller et al. [14] defined rules for confusion detection that are based on students' use of two types of emoji. They showed that the resulting rules closely align with the ground truth judgement of educational experts. We generalize both works to the more challenging task of recognizing multiple affects, and explore ways to facilitate this task by combining clustering methods with input from the course staff.

Felbo et al. [13] trained an LSTM architecture to predict emoji use in Twitter as proxies for users' emotional states. They show the model was able to generalize to other datasets containing self-reported emotional states. They employed clustering methods to learn relationships between 64 different emoji. We go beyond this work in several ways. First, we study the relationship between emoji use on the topic level. Second, we use the clusters to build better predictive models. Third, we involve course instructors in the analysis and use them to determine the best partition.

Çöltekin and Rama [10] used SVMs to predict emojis with a bag-of-b-grams feature set, combining both character n-grams and word n-grams and weighted by the TF-IDF score. This model achieved top performance in a recent competition for predicting 20 emoji on Twitter (SemEval 2018 task 2) [2]. Zhang et al. [30] used a BERT model for the emoji prediction task that outperformed Çöltekin and Rama [10] approach. We directly extend Zhang et al. model in adapting BERT to a biology course setting with an additional pre-training over the course's previous data.

3. THE NB SETTING

The Nota Bene (NB) web application is an open source social annotation tool that was developed at MIT [31]. NB is used in hundreds of university courses and includes more than 40,000 registered student users. The main feature of Nota Bene gives users the ability to directly annotate course content. Course content (PDF, HTML, or video file) is uploaded to the NB website by instructors. Students can annotate the content by highlighting a passage in the document (called "the marked text") and then add a post by typing into a text field that appears in the margin. These annotations may be used to create a post or to ask questions about the content. Classmates are encouraged to reply to other students' comments and to answer any posted questions.

NB posts are organized into threads, which consist of a starting comment or question followed by all the replies made by other students or instructional team to the initial annotation or to the subsequent replies. The in-place structure of the NB tool allows students to interact in the forum while they are reading the course material and provides context to the discussion. This structure has been shown to be beneficial for learning [22]. The NB interface allows students to express emotions and other affects in their comments via hashtags (which are translated to graphical representations of emoji), thus allowing students and instructors to filter

students' comments based on the type of affect in which they are interested.

3.1 The FYBIO course and emoji

FYBIO (First Year BIOlogy) is a general biology course required for all life sciences majors at a large, public university and is typically taken by students during their first year of study. Depending on the academic term it is offered, the course consists of 25 or 26 lectures. During this study, the course staff posted reading materials before each lecture and the students were assigned to read these materials and to provide three substantial posts in NB before each class. This encouraged active participation in forum discussions. Students received additional credit for including at least one emoji in at least one of their posts per lecture. The NB interface displays the hashtags graphically using relevant emoji symbols in students' posts. The emoji were designed by the course instructors to allow students the option to express emotions and opinions about course material, as well as invite assistance or participation from their peers or instructors. For the remainder of this paper, we will use the term "affects" to refer to all of the above expressions.

A full list of the emoji and their intended uses is shown in Table 1. Some examples of emoji and their intended uses are as follows: The #i-think emoji expresses an idea to share; the #lets-discuss emoji invites students and instructors to contribute to a discussion; the #learning-goal emoji identifies a topic or idea related to the course objectives; the #question emoji requests help from the course staff regarding a topic.

Emoji	Associated hashtag	Intended use	Percentage of use
?	#question	...you would like the professor to address a specific question in class.	18.7%
💭	#i-think	...you have an idea to share about something but are not sure if it is accurate.	18%
🤔	#just-curious	...you would like to learn more about a certain topic that you are curious about or intrigued by.	13.4%
!!	#important	...if you found a topic or idea to be important.	11.7%
❤️	#interesting-topic	...you found a topic or idea to be interesting.	10.7%
💡	#lightbulb-moment	...something (e.g., course material, a peer's post) led you to a new understanding (an "ah-ha" moment).	8.3%
🗣️	#lets-discuss	...you would like to invite your fellow classmates and instructors to weigh in, in a NB discussion.	5.6%
🌍	#real-life-application	...a topic or idea has any implications in the real world.	4.2%
😮	#surprised	...you found a topic or idea to be surprising.	3.9%
😞	#lost	...you are totally confused or overwhelmed about a topic and have no idea what question to even ask.	2.9%
🗨️	#learning-goal	...you identify a topic or idea related to the course objectives or reading-specific learning goals	2.5%

Table 1: Hashtags in NB, their associated emoji, intended uses and percentage of usage

Nota Bene provides instructors with a heatmap showcasing students' use of the different emoji (See Figure 1). Each emoji is displayed using a different color, and the brightness varies with respect to the density of the emoji in the posts.

Instructors can filter which emoji to display.

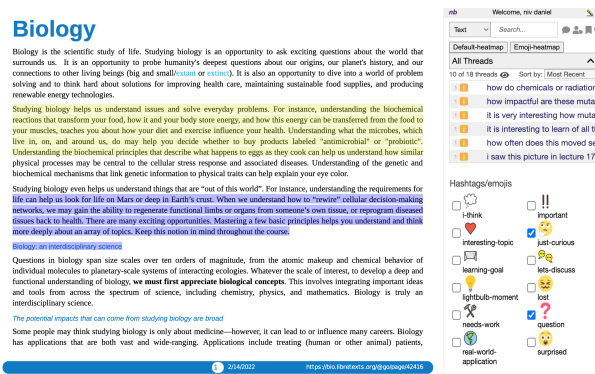


Figure 1: Emoji heat map in Nota Bene

Table 2: Statistics of our data set (different course instances of FYBIO)

Statistics (Num.)	Winter 2021	Summer 2021	Winter 2022	Total
Lectures	14	26	14	54
students	758	182	905	1,845
posts	28,072	15,072	36,165	79,309
emoji	23,867	10,635	28,674	62,906

Our dataset contained 79,309 unique student posts from three instances of the FYBIO course, as shown in Table 2. In total, 55,437 posts contained at least one emoji.

All students who participated in FYBIO courses filled a consent form allowing their data to be anonymized and analyzed for the sake of this study. The study was reviewed by the IRB (1456274-1) and deemed exempt. All students whose data was included in the study filled out a consent form and opted into the study. The high fraction of posts containing emoji (over 70%) shows that students' use of this tool went beyond the requirement in the course and echos past work demonstrating the educational benefits of affect [1, 18, 27].

4. METHODOLOGY AND RESULTS

Our goal is to provide teachers with better tools for making sense of students' affect in the course. To this end, we explore two main research questions. First, how do students use emoji in their posts? Second, can we use machine learning to infer how students would have tagged posts using the available emoji?

Our methodology addresses the first research question with a detailed analysis of emoji usage at the post and paragraph level, and addresses the second research question with the design of language models for emoji classification from text.

4.1 Analysis at the post level

Table 1 shows the percentage of emoji use in FYBIO. The table shows that the #question and #i-think emoji were used most often. Both of these emoji reflect uncertainty about the material and invite participation by instructors and students. The emoji expressing more direct requests for participation (e.g., #lets-discuss, #lost) were used much less frequently. This may reflect a resistance towards revealing

information about understanding that may have an adverse affect for students (e.g., peer pressure or getting a lower grade). Future editions of FYBIO will allow anonymous posting, which may change the way students use these posts.

By analyzing students’ comments we have found that about 70% of posts included at least one emoji. The majority (about 89%) of these posts include a single emoji, showing that students express a single affect in each post. The use of multiple emojis in the same post creates a complex affect. 10% of the posts contained two emoji and less than 1% of the posts contained 3 or 4 emoji.

4.2 Classifying individual emoji

In this section we describe how we use language models to classify students’ use of emoji at the post level. A good classification model can potentially aid instructors in making sense of students’ affective states in situations where emoji are not used, which is the case for most forums.

The model provides a mapping from a post to the most relevant emoji for the post according to the family. To this end we define a multiclass prediction task where the target class is the set of emoji shown in Table 1.

An instance was created for each post in the FYBIO dataset, and labeled with the relevant emoji for the post. For posts with multiple emoji, duplicate instances of the post were created for each emoji. The instances were split into training (85%) and test sets (15%), such that 15% randomly sampled instances of the training set are used as a validation set. We stratified the training and test sets such that each set contains approximately the same percentage of samples of each target class. In no case did the training and test sets include an instance with the same post. We compare two types of language models for the classification task. BERT is a pre-trained, transformer based language model that is commonly used in state-of-the-art natural language tasks [11]. We used an open-source BERT configuration (“bert-base-uncased”) that was trained on broad domain corpora (English Wikipedia and BooksCorpus).¹ The architecture contains 12 layers, 768 hidden units and 12 attention heads. We use this architecture in all of our experiments. Augmenting BERT with a pre-training procedure has demonstrated promising results in downstream NLP tasks [5, 20].

Our proposed model architecture (See Figure 2) consists of two components: 1) A language representation model (pre-training step) and 2) a fully connected neural network (fine tuning step). The language representation model was pre-trained using two sources: The FYBIO text book (150K words), as well as students’ posts from past instances of the course in 2020 (6.4M words). “bert-base-uncased” comes with a predefined vocabulary which consist of the approximately 30K most frequent words and sub-words from its pre-training corpus. We initiated the BERT model previously described, with its pre-trained weights, and afterwards trained the model for 50 additional epochs with a learning rate of $2e - 5$, which gave us the best results. Note that BERT allows the user to add special words or sub-words

¹<https://huggingface.co/transformers/modeldoc/bert.html>

that are unique to our domain. Since we pretrained using the Biology text book, we did not encounter words that are out of vocabulary, and so did not change the vocabulary.

The neural network was trained for the emoji classification task using 768 nodes with a softmax activation function above the additional pre-trained model. The input to this network was the embedded student’s post using the language representation model. The output of the network is a probability distribution over the 11 emoji in the target class. We fine-tuned the model with an addition of three epochs with a learning rate of $2e - 5$, using a maximal sentence length of 250, which gave us the best results.

We compared BERT to a bi-directional LSTM architecture, similar to the one used by Felbo et al. and Baziotis et al. [13, 4] to classify emoji in social media. The model included five layers. One layer was used to embed words in students’ posts as high-dimensional vectors; three layers for classification consisting of 64 LSTM units (32 units in each direction); the final layer was an attention mechanism layer that connects words in posts with preceding and succeeding words while computing the importance of each word with the corresponding label. The model was implemented using Python’s keras package [9]. We used a separate pre-training process using word2vec [21] to construct a high dimensional 200-sized vector representation of words in students’ posts. The pre-training used the FYBIO textbook, as well as students’ posts from 2020 and implemented using Python’s Gensim package [24].

Table 3 compares the BERT and LSTM models when classifying emoji according to precision, recall, and weighted F-1 scores. We can see that the BERT model outperforms the Bi-LSTM model in all three metrics by a significant margin (McNemar’s test, $p < 1.16 \cdot e^{-35}$). We attribute this difference to the pre-trained language model in BERT which allows it to generalize to domains with low amounts of training data [11].

Table 3: Model output comparison

Model	Precision	Recall	F-1 Score
BERT	40.2%	43.3%	40.7%
LSTM	26.6%	32.8%	29.3%

We note that prior work using BERT to classify emoji report a macro F-1 score of 38.5% [30], while our Macro-F1 score is lower (32.2%). However, we do not compare directly with these models for several reasons. First, they used an order of magnitude more data (550K tweets vs. 50K posts). Second, their target set was larger (20 vs. 11 emoji). Third, the emoji setting is different (e.g., smiley-faces, hearts, etc.) and is used in different ways (e.g., use the same emoji in different parts of the sentence).

Table 4 breaks down the performance of the model according to individual emoji. The table shows a positive relationship between the amount of training data for a given emoji and the prediction performance of the model for the emoji. An interesting exception is the #lets-discuss emoji, which is pedagogically important and represents more than 6% of the dataset, but achieves very low performance. The reason

Figure 2: Model architecture used for Emoji prediction

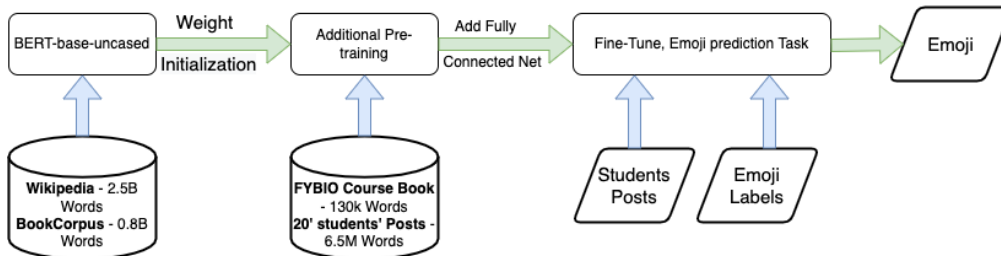


Table 4: BERT prediction score, divided by prediction output score for each emoji

Metric	?	☁️	😬	!!	❤️	💡	💬	🌍	😮	😞	🗨️
Precision	0.55	0.49	0.37	0.4	0.33	0.33	0.18	0.31	0.33	0.39	0.19
Recall	0.72	0.62	0.28	0.52	0.34	0.26	0.05	0.23	0.28	0.2	0.05
F-1 Score	0.62	0.55	0.32	0.45	0.34	0.29	0.08	0.26	0.3	0.26	0.07

is that the model misclassified over 50% of #lets-discuss as #i-think, suggesting that students may use these two emoji interchangeably to convey the same affect. We study this question in the next section.

Another possible reason for the low performance for some of the emoji may be that they were used by students to convey different affects than those intended for the emoji. For example, the following post shows a student has reached a conclusion but does not understand why it is true. This reflects a question rather than total confusion which was the originally intended purpose of this emoji: *I also get the impression that not having similar proof-reading mechanisms in transcription means that they are less severe. However. I'm not sure why this would be the case #lost.*

4.3 Classifying groups of emoji

In this section, we exploit the fact that emoji are often used together in paragraphs to cluster the emoji into pedagogically meaningful categories and predict the categories, rather than individual emoji. Felbo et al [13] clustered emoji based on the relationships between predicted probabilities of a computational model. We follow their approach and apply a hierarchical clustering algorithm with average linkage on the correlation matrix on the predicted probabilities of the BERT model shown in Table 4. Each emoji is represented as a vector of predicted probabilities for each post in the test set. To illustrate this approach, consider the following post from the test set: *is there an operon for every single metabolite? [trp] operon applies to tryptophan while lac operon applies to lactose. how many different types of operons are there and how they function differently?* The BERT model, applied to this post, outputted the highest probability for the #question and #just-curious emoji (0.467 and 0.45 probability, respectively) and low probability for the rest of the emoji (e.g., 0.008 probability for the #interesting-topic emoji, and 0.003 probability for the #lightbulb-moment emoji). Note that because we are using the softmax function, these probabilities add up to 1, although the two emoji have a combined probability of approximately 0.92. The fact that the model is likely to assign a high or low probability to both #question and #just-

curious emoji for a given post suggests that they should form a single category.

Figure 3 shows the dendrogram that is outputted by applying hierarchical clustering. The height of each node in the y -axis is proportional to the value of the intergroup dissimilarity between its child nodes. The distance threshold of the clustering is used to determine which nodes to put in the same cluster.

In concurrence with the FYBIO instructions, we determined a threshold of 1. This threshold grouped the 11 emoji into three emoji pair clusters and 5 singleton emoji clusters: (#question, #just-curious), (#lets-discuss, #i-think), (#important, #learning-goal), (#surprised), (#lightbulb-moment), (#lost), (#interesting-topic), and (#realworld-application).

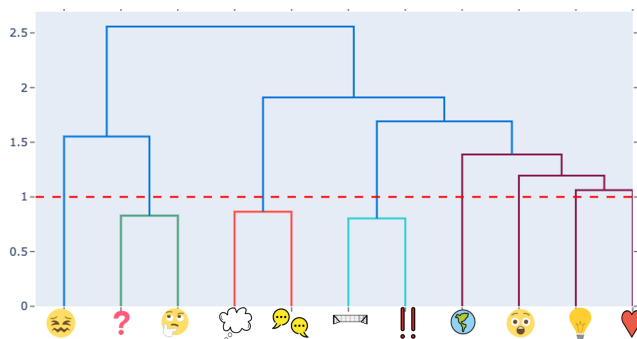


Figure 3: Hierarchical clustering of the emoji based on the model's prediction. The dashed red line represents the confidence threshold.

The justification provided by the course instructors for this division are as follows: With respect to the cluster containing the emoji (#question, #just-curious), instructors claimed that they invite a response from peers or the instructors, remarking that *“They are both requesting responses but with different urgency.”* The individual emoji were also highly correlated on the paragraph level. Both of these aspects led instructors to agree to put them in the same category.

Table 5: Model score of each cluster and the weighted average score of all labels

Cluster	Precision	Recall	F-1 Score
#question, #just-curious	77.8%	83.9%	80.7%
#lets-discuss, #i-think	58.4%	60.5%	59.5%
#important, #learning-goal	47.3%	51.4%	49.3%
Total Score (8 Categories)	55.2%	56.8%	55.8%

With respect to the cluster containing the emoji (#lets-discuss, #i-think), instructors claimed that *“both of these emoji indicate enthusiasm to continue around a topic, either for curiosity or sometimes to clarify [...] they are expressing how they understand it and want clarification or alternate views from peers.”* The individual emojis were also highly correlated on the paragraph level.

With respect to the cluster with the emoji (#important, #learning-goal), instructors claimed that both emoji depict students’ perspectives on exam-related content. They claimed that *“these emojis are used when students identify parts of the text that they believe are important for them to perform well in the course - that may be linked to assessment or the development of knowledge that build towards good performance on assessments.”*

Interestingly, the emoji-pair (#surprised, #interesting-topic) exhibited a high correlation on the paragraph level but was deemed sufficiently distinct by instructors to warrant a separate affect category for each emoji. Instructors claimed that *“Both emoji are expressions of ”enjoyment” of the text - though as noted slightly different.”*

The last part of our methodology was to study whether predicting the 8 categories, rather than the 11 original emojis, would improve prediction performance. To this end we trained the BERT model with the same architecture as described in Section 4.2, using the 8 categories as the target set. Table 5 shows the performance for the three emoji-pair clusters as well as total performance over all 8 of the categories. As shown by the Table, there was an improvement of 15% in prediction performance (from 40% to 55.8%) when compared to using a target set containing the original 11 emoji. We note that an improvement in prediction performance was to be expected, given the reduction in the size of the target set. What is interesting is that the two emoji #lets-discuss and #learning-goal, which achieved very low performance when predicted as individual emoji (see Table 4), exhibited significant improvement when clustered together with another emoji (F-Score of 59.5% and 49.3% respectively). The cluster with the emoji pair (#question, #just-curious) received the greatest score across all metrics. In terms of predicting the clusters with a single emoji, we saw a slight reduction in performance (less than one percent).

5. DISCUSSION AND CONCLUSIONS

This paper studied students’ use of emoji in a large scale course forums used by hundreds of students. We found that when made available to them, students tend to use emoji in their posts to enhance their meaning or to express affect (an emotion, belief and opinion), rather than as a substitute

for words. The most popular use of emoji was to invite further explanation or to express an interest in a given topic. Emoji expressing confusion or misunderstandings were less popular, and may reflect students’ hesitation to expose these affects in public. Students’ use of emoji is complex. In some cases, students use multiple emojis in the same post to convey a joint affect, while in other cases students use different emoji interchangeably to mean the same affect. Some students may use emoji in a different way than their intended use, and instructors may wish to present their meaning to students in the beginning of the course.

We began to explore the use of emoji as a pedagogical tool by instructors to aid course design or guide students. To this end, we analyzed how topics in the course material generate different emoji reactions from students. This led us to partition emoji into groups using hierarchical clustering. We identified the most pedagogically meaningful emoji clusters with the help of the course instructors, and designed a language model based on BERT that was able to classify students’ posts to the right cluster with good performance.

The language model allows to classify students’ posts with affects even when emoji are absent from the post, which can naturally extend our contributions to other courses and forums. By adding more data to the learning process, we hope to improve our language model. Following the improvement, we want to apply the model in two separate scenarios. (1) non-bio NB course (with same emojis) (2) course forum with different labels in hope that the model is able to learn affects signals from students writes.

We also intend to use our insights and computational tools to help teachers make sense of student affect in an active class. We envision a dashboard that would alert instructors to affects they care about, like confusion, or insights conveyed by students about learning goals. We wish to study how teachers use this tool to inform their course design, or to actively intervene in a forum to guide discussion.

Acknowledgements

Part of this work was carried out thanks to a grant by the National Science Foundation #1915724 and EU Horizon 2020 Grant Agreement number 823783.

6. ADDITIONAL AUTHORS

Jumana Almahmoud (MIT, email: jumanam@mit.edu and Michele Igo (U.C. Davis, email: mmigo@ucdavis.edu)

7. REFERENCES

- [1] R. S. Baker, S. K. D’Mello, M. M. T. Rodrigo, and A. C. Graesser. Better to be frustrated than bored: The incidence, persistence, and impact of learners’ cognitive-affective states during interactions with three different computer-based learning environments. *International Journal of Human-Computer Studies*, 68(4):223–241, 2010.
- [2] F. Barbieri, J. Camacho-Collados, F. Ronzano, L. Espinosa-Anke, M. Ballesteros, V. Basile, V. Patti, and H. Saggion. SemEval 2018 task 2: Multilingual emoji prediction. In *Proceedings of The 12th International Workshop on Semantic Evaluation*,

- pages 24–33, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.
- [3] C. Baziotis, N. Athanasiou, G. Paraskevopoulos, N. Ellinas, A. Kolovou, and A. Potamianos. NTUA-SLP at semeval-2018 task 2: Predicting emojis using rnns with context-aware attention. *CoRR*, abs/1804.06657, 2018.
 - [4] C. Baziotis, N. Athanasiou, G. Paraskevopoulos, N. Ellinas, A. Kolovou, and A. Potamianos. Ntua-slp at semeval-2018 task 2: Predicting emojis using rnns with context-aware attention. *arXiv preprint arXiv:1804.06657*, 2018.
 - [5] I. Beltagy, K. Lo, and A. Cohan. Scibert: A pretrained language model for scientific text. *arXiv preprint arXiv:1903.10676*, 2019.
 - [6] H. H. Binali, C. Wu, and V. Potdar. A new significant area: Emotion detection in e-learning using opinion mining techniques. In *2009 3rd IEEE International Conference on Digital Ecosystems and Technologies*, pages 259–264, 2009.
 - [7] M. Boekaerts and R. Pekrun. Emotions and emotion regulation in academic settings. In *Handbook of educational psychology*, pages 90–104. Routledge, 2015.
 - [8] D. S. Chaplot, E. Rhim, and J. Kim. Predicting student attrition in moocs using sentiment analysis and neural networks. In *CEUR Workshop Proceedings*, volume 1432, pages 7–12, 2015.
 - [9] F. Chollet et al. Keras. <https://keras.io>, 2015.
 - [10] Ç. Çöltekin and T. Rama. Tübingen-oslo at semeval-2018 task 2: Svms perform better than rnns in emoji prediction. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 34–38, 2018.
 - [11] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
 - [12] M. L. B. Estrada, R. Z. Cabada, R. O. Bustillos, and M. Graff. Opinion mining and emotion recognition applied to learning environments. *Expert Systems with Applications*, 150:113265, 2020.
 - [13] B. Felbo, A. Mislove, A. Søgaard, I. Rahwan, and S. Lehmann. Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. *arXiv preprint arXiv:1708.00524*, 2017.
 - [14] S. A. Geller, N. Hoernle, K. Gal, A. Segal, A. X. Zhang, D. Karger, M. T. Facciotti, and M. Igo. # confused and beyond: detecting confusion in course forums using students’ hashtags. In *Proceedings of the Tenth International Conference on Learning Analytics & Knowledge*, pages 589–594, 2020.
 - [15] R. Jena. Sentiment mining in a collaborative learning environment: capitalising on big data. *Behaviour & Information Technology*, 38(9):986–1001, 2019.
 - [16] Z. Kastrati, F. Dalipi, A. S. Imran, K. Pireva Nuci, and M. A. Wani. Sentiment analysis of students’ feedback with nlp and deep learning: A systematic mapping study. *Applied Sciences*, 11(9):3986, 2021.
 - [17] L. Kerslake and R. Wegerif. The semiotics of emoji: the rise of visual language in the age of the internet (book review). *Media and Communication*, 5(4):75–78, 2017.
 - [18] B. Kort, R. Reilly, and R. W. Picard. An affective model of interplay between emotions and learning: Reengineering educational pedagogy-building a learning companion. In *Proceedings IEEE international conference on advanced learning technologies*, pages 43–46. IEEE, 2001.
 - [19] P. Kralj Novak, J. Smailović, B. Sluban, and I. Mozetič. Sentiment of emojis. *PLoS ONE*, 10(12):e0144296, 2015.
 - [20] J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So, and J. Kang. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240, 2020.
 - [21] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.
 - [22] K. Miller, S. Zyto, D. Karger, J. Yoo, and E. Mazur. Analysis of student engagement in an online annotation system in the context of a flipped introductory physics class. *Physical Review Physics Education Research*, 12(2):020143, 2016.
 - [23] Z. A. Pardos, R. S. Baker, M. O. San Pedro, S. M. Gowda, and S. M. Gowda. Affective states and state tests: Investigating how affect and engagement during the school year predict end-of-year learning outcomes. *Journal of Learning Analytics*, 1(1):107–128, 2014.
 - [24] R. Rehurek and P. Sojka. Gensim–python framework for vector space modelling. *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic*, 3(2), 2011.
 - [25] G. Santamaría-Bonfil and O. G. T. López. Emoji as a proxy of emotional communication. In *Becoming Human with Humanoid-From Physical Interaction to Social Intelligence*. IntechOpen, 2019.
 - [26] R. Sylwester. How emotions affect learning. *Educational leadership*, 52(2):60–65, 1994.
 - [27] G. J. Trevors, K. R. Muis, R. Pekrun, G. M. Sinatra, and M. M. Muijselaar. Exploring the relations between epistemic beliefs, emotions, and learning from texts. *Contemporary Educational Psychology*, 48:116–132, 2017.
 - [28] M. Wen, D. Yang, and C. Rose. Sentiment analysis in mooc discussion forums: What does it tell us? In *Educational data mining 2014*. Citeseer, 2014.
 - [29] A. X. Zhang, M. Igo, M. Facciotti, and D. Karger. Using student annotated hashtags and emojis to collect nuanced affective states. In *Proceedings of the Fourth (2017) ACM Conference on Learning@ Scale*, pages 319–322, 2017.
 - [30] L. Zhang, Y. Zhou, T. Erekhinskaya, and D. Moldovan. Emoji prediction: A transfer learning approach. In K. Arai, S. Kapoor, and R. Bhatia, editors, *Advances in Information and Communication*, pages 864–872, Cham, 2020. Springer International Publishing.
 - [31] S. Zyto, D. Karger, M. Ackerman, and S. Mahajan. Successful classroom deployment of a social document annotation system. In *Proceedings of the sigchi conference on human factors in computing systems*, pages 1883–1892, 2012.

Investigating Growth of Representational Competencies by Knowledge-Component Model

Jihyun Rho (jrho6@wisc.edu)¹, Martina A. Rau (marau@wisc.edu)¹, Barry D. Van Veen²

Department of Educational Psychology¹, Department of Electrical and Computer Engineering²
University of Wisconsin-Madison

ABSTRACT

Instruction in many STEM domains heavily relies on visual representations, such as graphs, figures, and diagrams. However, students who lack representational competencies do not benefit from these visual representations. Therefore, students must learn not only content knowledge but also representational competencies. Further, as learning progresses, knowledge likely becomes more abstract, so that content knowledge may no longer be tied to a specific representation. This raises the question of whether students integrate representational competencies with content knowledge as learning progresses. The present study addresses this question by building knowledge-component models using log data collected from two studies in an introductory electrical engineering course. We compared knowledge-component models that separate representational competencies from content knowledge to knowledge-component models that integrate representational competencies with content knowledge. Our results show that as learning progressed, integrated knowledge-component models had better model fit. This finding indicates that over time, students' representational competencies become gradually integrated into content knowledge. Further, this suggests that different knowledge-component models might be needed at different times during a learning progression.

Keywords

Representational competencies, content knowledge, knowledge-component model, intelligent tutoring system

1. INTRODUCTION

The success of adaptive educational technologies depends on analyses of students' knowledge growth during their interaction with problem-solving activities. These analyses equip the educational technology with information about the students' current learning progress to mastery of the targeted knowledge [32] and enables it to provide adaptive feedback or to select appropriate interventions [16] based on the individual student's learning progress [7]. This capability has contributed to the success of adaptive educational technologies [30].

Therefore, much research has investigated how to analyze students' knowledge growth based on log data generated by students' problem-solving interactions in educational technologies. The first step in analyzing knowledge growth is to capture students' knowledge in a way that can then be used to trace their knowledge acquisition over time [4, 7]. Knowledge-component models are a common way of capturing students' knowledge [15]. The basic assumption of

knowledge-component models is that knowledge consists of fine-grained "atom-like" components [15]. Hence, knowledge-component modeling seeks to identify all knowledge components required for mastering the targeted knowledge [13].

Traditional knowledge-component models have focused on capturing content knowledge. However, focusing on only content knowledge may not adequately enable educational technologies to support students' learning, especially in STEM fields. Previous research showed that, students often have difficulties understanding visual representations, while most STEM instruction heavily relies on multiple visual representations [2, 17]. Such struggles can impede their content learning [24]. For example, electrical engineering courses on signal processing frequently use visual representations as shown in Fig. 1 to explain concepts related to sinusoids. While visual representations are often thought to support learning [1], they can impede learning for students who do not know how to interpret the visual representations. For instance, if the students are unfamiliar with time-domain graphs (Fig. 1a) or phasor graphs (Fig. 1b), they may struggle to understand the concept of the sinusoid. This example typifies that many instructional scenarios expect students to have *representational competencies*. Representational competencies are defined as the knowledge and skills that enable students to understand and use visual representations to reason and solve tasks [9].

While most research on knowledge-component models has focused on content knowledge, only a few studies show that capturing representational competencies in addition to content knowledge improves the fit of knowledge-component models [25]. A limitation of these studies is that they have assumed a static structure of knowledge-component models; that is, representational competencies and content knowledge were captured as separate knowledge components, and this did not change over time. However, as learning progresses, students' content knowledge likely becomes more abstract and their use of representational competencies becomes more automated. Thus, the goal of this paper is to address this limitation by comparing knowledge-component models that separate or integrate representational competencies and content knowledge in various ways.

2. LITERATURE REVIEW

In the following, we first review research on representational competencies. Then, we briefly review the few prior studies that have captured representational competencies in knowledge-component models.

2.1 Representational Competencies

The educational psychology has identified several types of representational competencies that enable students to learn content knowledge from visual representations [24].

First, students need *visual-understanding competencies*: the ability to map visual features to relevant to-be-learned content [28]. In the

J. Rho, M. Rau, and B. Vanveen. Investigating growth of representational competencies by knowledge-component model. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 346–352, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853077>

previously mentioned example of a student learning about sinusoids, visual-understanding competencies allow the student to map a visual feature (e.g., the peak in the time-domain graph) to the concept it depicts (e.g., the amplitude of a sinusoid).

Second, students need *conceptual connection-making competencies*: the ability to conceptually understand similarities and differences between multiple visual representations [1]. This allows students to explain how domain-relevant concepts are depicted in different visual representations [29]. For example, the red arrow in Fig. 1 illustrates how a student should connect the amplitude in the time-domain graph (Fig. 1a) to the phasor's magnitude in the phasor graph (Fig. 1b). Conceptual connection-making competencies also involve identifying visual features that have surface similarities among visual representations but are conceptually irrelevant [11]. For example, a student may notice that both the time-domain graph (Fig. 1a) and the phasor graph (Fig. 1b) have two axes. Yet, the axes represent different concepts: time and amplitude in the time-domain graph, the imaginary and real parts of the phasor in the phasor graph.

Third, students need *perceptual connection-making competencies*: the ability to effortlessly and efficiently translate between multiple visual representations [12]. Students with perceptual connection-making competencies can intuitively translate between two visual representations and quickly judge whether they depict the same concept, without experiencing mental effort when executing this task [12]. For example, a perceptually proficient student would see "at a glance" the phasor graph in Fig. 1b represents the amplitude of the sinusoid in Fig. 1a.

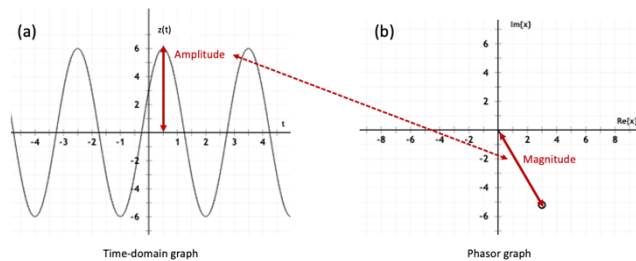


Figure 1. Mapping time-domain graph amplitude (a) to phasor graph magnitude (b).

2.2 Knowledge-Component Model

Adaptive educational technologies require information about the students' learning progress in order to provide individualized support. [30]. They must describe the knowledge students have already learned and what knowledge they still have to learn [6]. Furthermore, adaptive educational technologies rely on algorithms that predict which types of support (e.g., contextual feedback, choice of problem-solving activities) would help the student acquire the knowledge s/he has not yet learned [21].

Educational technologies rely on knowledge-component models to trace students' knowledge acquisition and to predict the growth of students' knowledge. Knowledge-component models represent "acquired units of cognitive function that can be inferred from performance on a set of related tasks" [15]. Here, knowledge components refer to a unit of distinct skills or concepts, which together compose the knowledge students learn in problem-solving activities. Therefore, the accuracy of a knowledge-component model depends on identifying all relevant knowledge components that describe the targeted knowledge [13, 16].

Cognitive Task Analysis (CTA) is one prominent method to describe the requisite knowledge components to perform a task [33]. However, since CTA relies on a thorough analysis of how experts solve tasks, it is time consuming. To increase the efficiency of knowledge-component modeling, educational data mining techniques can be used to automate the process of building models, including learning factors analysis (LFA) [4], Knowledge Spaces [31], and matrix factorization [8]. Typically, multiple potential models are compared using Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC) scores [14]. However, since this research has focused mostly on modeling content knowledge, the interplay between representational competencies and content-focused knowledge components remains unexamined.

One study compared knowledge-component models that captured (1) only content knowledge and (2) content knowledge and representational competencies. Capturing both content knowledge and representational competencies resulted in better model fit [25]. A follow-up study tested whether adapting instructional support to students' representational competencies in addition to content knowledge resulted in higher learning outcomes than adapting only to content knowledge [23]. Results showed that adapting to students' representational competencies in addition to content knowledge enhanced students' learning of content knowledge.

However, this prior research is limited in two ways. First, while most prior research focuses on modeling content knowledge [18, 20], the few studies that have also modeled representational competencies [23, 25] have focused on a particular domain; namely chemistry. Therefore, we seek to replicate these findings in another domain. A second limitation is that the prior studies assumed that the structure of the knowledge-component model remains static over time. Yet, according to expert-novice research, students gradually acquire highly abstract schemas about the content knowledge relevant to a given domain [5, 10]. This yields the hypothesis that capturing representational competencies separately from content knowledge is most important early in a learning progression whereas later in a learning progression, representational competencies likely become integrated with content knowledge.

3. HYPOTHESES

To address the limitations of prior research just described, we test:

Hypothesis 1: A knowledge-component model that captures representational competencies and content knowledge is more accurate than a knowledge-component model that captures only content knowledge or a knowledge-component model that captures only representational competencies.

Hypothesis 2: As students' learning progresses, a knowledge-component model that integrates content knowledge and representational competencies is more accurate than a knowledge-component model that captures content knowledge separately from representational competencies.

4. DATASETS

To test these hypotheses, we use log data generated from students' problem-solving interactions in Signals Tutor, an intelligent tutoring system (ITS) for undergraduate electrical engineering. In the following, we first describe the problem-solving activities in Signals Tutor, and then the log data we used to test our hypotheses.

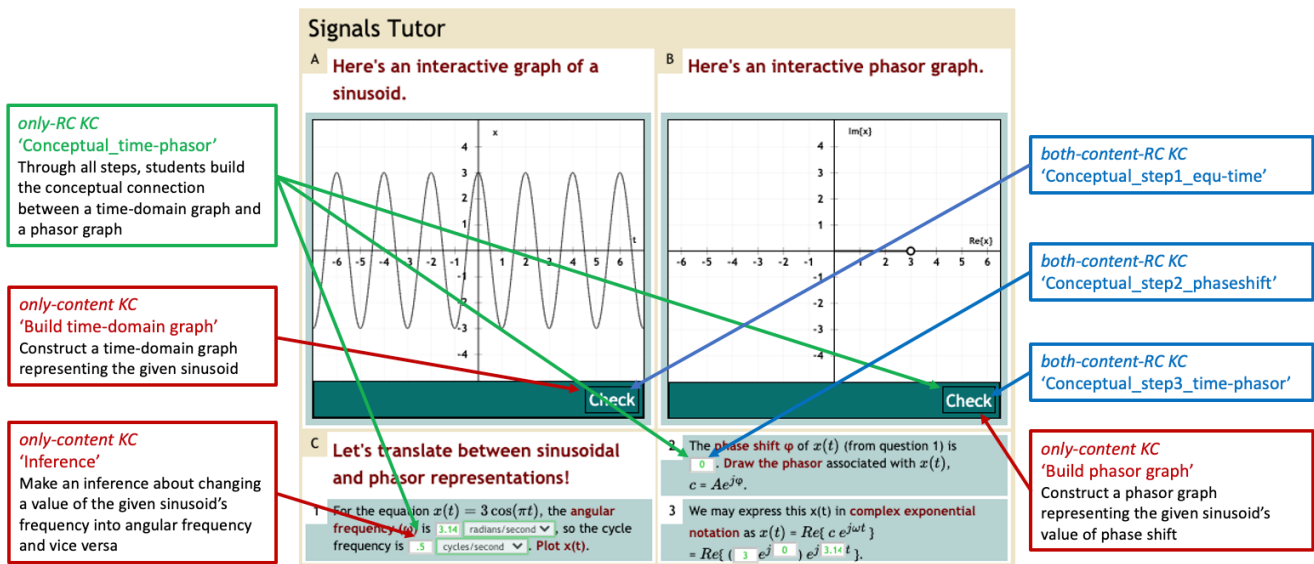


Figure 2 Example of a problem in Signals Tutor and steps that labelled with different knowledge components for hypothesis 1

4.1 Signals Tutor

Signals Tutor supports learning through problem solving [22, 27, 30]. As is typical for ITSs, Signals Tutor provides step-by-step guidance for complex problem-solving tasks [30], detects multiple possible solution paths, provides personalized feedback that addresses diagnosed misconceptions, and on-demand hints for each step. As illustrated in Fig. 2, students work with interactive visual representations to visually depict concepts related to sinusoids.

Because Signals Tutor incorporates multiple visual representations, it offers opportunities for students to practice representational competencies. Specifically, students practice *visual-understanding* competencies when they interact with one visual to make sense of sinusoid concepts. Students practice *conceptual connection-making* competencies when they have to integrate information from multiple visual representations to understand sinusoid concepts. They practice *perceptual connection-making* competencies when a task requires translating quickly among multiple visual representations to extract relevant information about sinusoids.

4.2 Log Data

We collected log data from two studies that were conducted as part of an introductory electrical engineering course on signal processing at a university in the Midwestern U.S. Study 1 was conducted in Fall 2020; Study 2 was conducted in Spring 2021. The course was taught online during both semesters. Students used Signals Tutor as part of the course for a study described elsewhere [26]. The present paper focuses on log data generated from students' interactions with two units of Signals Tutor that provided opportunities to practice the representational competencies described above. Specifically, we extracted 84,960 transactions generated by 136 students from Study 1 log data and 76,786 transactions generated by 145 students from Study 2. These transactions involved problem-solving steps where students constructed visual representations and equations and answered conceptual questions by selecting answers from drop-down menus or via text input.

5. ANALYSIS

To test hypothesis 1, we created knowledge-component models with and without representational competencies. To test hypothesis 2, we created several knowledge-component models that captures

the integration of content knowledge and representational competencies. We compared the fit of each model in the earlier vs. the later unit of Signals Tutor.

5.1 Knowledge-Component Models with and without Representational Competencies

To test hypothesis 1, we created three knowledge-component models: (1) the *only-content-KC model* captures only content knowledge, (2) the *only-RC-KC model* captures only representational competencies, and (3) the *both-content-and-RC-KC model* captures both content knowledge and representational competencies. The knowledge components captured by each model were derived from manual cognitive task analysis relying on expert content knowledge provided by an engineering professor who taught the course for more than 30 years/decades.

First, the *only-content-KC model* contains 9 *only-content knowledge components* that describe concepts and skills irrespective of the representational competencies, listed in Table 1. For example, the knowledge component 'Inference' in Table 1 refers students' ability to make an inference about changing a value of the given sinusoid's frequency into the angular frequency.

Second, the *only-RC-KC model* captures only representational competencies but not content knowledge. It contains 11 knowledge components, listed in Table 1. As mentioned above, Signals Tutor offers opportunities to practice three types of representational competencies. The *only-RC-KC model* describes the competencies students need to understand the visual representations used in the problems; for example, conceptual connection-making competencies (e.g., 'Conceptual_time-phasor') and perceptual connection-making competencies (e.g., 'Perceptual_time-phasor') related to translating a time-domain graph to a phasor graph.

Finally, the *both-content-and-RC-KC model* captures both content knowledge and representational competencies. It contains 42 knowledge components, listed in Table 1. To develop content knowledge of translating a time-domain to a phasor graph, students practice competencies for making both conceptual and perceptual connections among visuals. For example, consider the following steps that provided practice opportunities for conceptual connection-making competencies. Students first built a time-domain graph

Table 1 Examples of knowledge components in Signals Tutor (For the value in [], students type in their answer).

Knowledge Components	Examples of knowledge components	Description	Example
<i>only-content KC</i>	Inference	Make inferences about changing the value of frequency to angular frequency or vice versa.	$x(t)$ is has an angular frequency ω of $\frac{\pi}{2}$ in radians/sec. The frequency in cycles/sec is [.25].
	Planning	Plan how to represent the sinusoid using different type of visual representation.	The complex amplitude can be represented visually by a [phasor].
	Build phasor graph	Construct a phasor graph of a given sinusoid	Plot the phasor corresponds to $x(t)$.
	Build time-domain graph	Construct a time-domain graph of a given sinusoid	Plot this sinusoid on the given time-domain graph.
<i>only-RC KC</i>	Conceptual_time-phasor	Make sense of how a time-domain visual correspond to a given phase-domain visual	The given graph shows a sinusoid $x(t)$. Plot the phasor corresponding to $x(t)$.
	Perceptual_phasor-time	Translate a phase-domain visual to a time-domain visual	Here's a phasor. Which cosine function represents that phasor?
<i>both-content-and-RC KC</i>	Conceptual_step1_equ_time	Given a cosine function, build a time-domain graph representing the given time-domain equation	For the equation $x(t) = 3\cos(\pi t)$, plot $x(t)$.
	Conceptual_step2_phaseshift	After building a time-domain graph in step 1, identify its value of phase shift	The phase shift ϕ of $x(t)$ is [0].
	Conceptual_step3_time-phasor	Based on identified information in step 2, translate a time-domain graph to a phasor graph	Draw the phasor associated with $x(t)$, $c = Ae^{j\phi}$.
	Conceptual_step4_phasor-exp	After building a phasor graph in step 3, write corresponding complex exponential notation.	We may express $x(t)$ in complex exponential notation as $Re\{[3]\exp(j[0])\exp(j[3.14]t)\}$
	Perceptual_time-phasor (clockwise)	Translate a time-domain graph to a phasor graph (rotated in the clockwise direction)	Here's a sinusoid (cosine function). Which phasor represents that sinusoid?
	Perceptual_time-phasor (counter-clock)	Translate a time-domain graph to a phasor graph (rotated in the counter-clockwise direction)	Here's a sinusoid (cosine function). Which phasor represents that sinusoid?

representing a given sinusoid represented in the equation form (e.g., 'Conceptual_step1_equ-time' KC in Table 1). The next step is to find the value of phase shift (e.g., 'Conceptual_step2_phaseshift'), which is basis for building a corresponding phasor graph in the next step. In the third step, students translate the time-domain graph to a phasor graph (e.g., 'Conceptual_step3_time-phasor').

Hypothesis 1 predicts that the *both-content-and-RC-KC model* has better model fit than the other knowledge-component models.

5.2 Knowledge-Component Models with and without abstracted Knowledge Components

To test hypothesis 2, we created two knowledge-component models: (1) the *separate-RC-KC model* describes the knowledge structure of students starting at the novice-level (2) the *integrated-RC-KC model* describes the knowledge structure of students reaching to the expert-level through working on Signals Tutor, detailed in the following.

First, based on learning gains we observed between units [26], we assumed that structural changes in knowledge components would occur between units 1 and 2. Therefore, we chose to investigate knowledge components that were common to units 1 and 2 and examined how these knowledge components changed after finishing each unit. Thus, starting with the list of knowledge components in the *both-content-and-RC-KC model* described in 5.1, we identified 33 knowledge components that unit 1 and unit 2 had in common.

This yielded the *separate-RC-KC model*. Given that this knowledge-component model separately captures content knowledge and representational competencies, we anticipate that this might capture the knowledge structure of novice students in the first unit.

Second, based on previous studies' finding that sufficient training makes students' knowledge become abstracted [5, 10] or abstracted away from the type of representational competencies [25], we assumed that students' knowledge components started to be integrated into content knowledge after finishing unit 1. Thus, we identified lists of knowledge components from the *both-content-and-RC-KC model* that describe similar content knowledge. For example, in Signals Tutor, students learn about concepts related to the phase shift of sinusoids, which we classify as content knowledge. Depending on how students interact with the visual representations that depict these concepts, they practice different representational competencies. The *both-content-and-RC-KC model* contains six knowledge components related to phase shift, illustrated in Table 1: three knowledge components describe students' ability to conceptually connect a sinusoid's shifted amount and direction shown in a time-domain graph to a phasor's rotational direction and amount shown in a phasor graph (e.g., 'Conceptual_step1_equ_time', 'Conceptual_step2_phaseshift', 'Conceptual_step3_time-phasor'), and two knowledge components describe students' ability to make perceptual connections between a time-domain graph and a phasor graph by treating the representations holistically (e.g., 'Perceptual_time-phasor (0)',

‘Perceptual_time-phasor (clockwise)’, ‘Perceptual_time-phasor (counter-clock)’).

If these six representational competencies become more integrated with content knowledge about phase shift after practice (hypothesis 2), separating these representational competencies from content knowledge may no longer adequately describe students’ abstracted knowledge structure. To capture this more abstracted understanding, we built new knowledge-component models that contained merged knowledge components. Starting with the knowledge components from the *separate-RC-KC model*, we first merged knowledge components that covered similar content knowledge. For example, ‘Conceptual_step1_equ_time’, ‘Conceptual_step2_phaseshift’, ‘Conceptual_step3_time-phasor’ and ‘Perceptual_time-phasor (clockwise)’ were merged into the ‘Build phasor graph’ knowledge component, which is one of the knowledge components from the *only-content KC model*. We then tested whether merging these knowledge components improved model fit using AIC and BIC scores. If it did, we kept the merged knowledge component; if it did not, we kept the separated knowledge components. We repeated these steps until there were no more opportunities to merge knowledge components. We carried out these steps separately for the data from Studies 1 and 2, yielding *integrated-RC-KC model-1* and *integrated-RC-KC model-2* as shown in shown in Table 2.

Hypothesis 2 predicts that the *integrated-RC-KC model-1* and the *integrated-RC-KC model-2* have a better model fit compared to the *separate-RC-KC model* in unit 2. Similarly, since students are novice at the beginning stage, hypothesis 2 predicts that the *separate-RC-KC model* shows better model fit compared to the *integrated-RC-KC model-1* and the *integrated-RC-KC model-2* in unit 1.

Table 2. Separate and integrated knowledge components.

Separate KCs in unit 1	Integrated KCs in unit 2
Conceptual_step3_time-phasor	Build phasor graph (<i>integrated-RC-KC model-1</i>)
Perceptual_time-phasor (0)	
Perceptual_time-phasor (clockwise)	
Perceptual_time-phasor (counter-clockwise)	
Perceptual_phasor-exp	Write complex exponential notation (<i>integrated-RC-KC model-1</i>)
Conceptual_step4_phasor-exp	
Perceptual_phasor-cartesian	Write cartesian form (<i>integrated-RC-KC model-2</i>)
Individual_phasor-cartesian (real part)	
Individual_phasor-cartesian (imaginary part)	

6. RESULTS

To test hypothesis 1, we compared the model fit of the *only-content-KC model*, *only-RC-KC model*, and *both-content-and-RC-KC model* using data from Studies 1 and 2. Table 3 shows that the *both-content-and-RC-KC model* has a better model fit than the *only-content-KC model* and the *only-RC-KC model* in Study 1 and Study 2. For AIC scores, the lower AIC indicate a better-fit-model, and more than -2 is considered significantly better than model it is being compared [3]. Similarly, a decrement greater than 10 indicates very

strong evidence in terms of BIC [19]. Lower values of RMSE also indicate better fit. These results support hypothesis 1.

To address hypothesis 2, we compared the model fit of the *separate-RC-KC model*, the *integrated-RC-KC model-1*, and the *integrated-RC-KC model-2* as shown in Table 4. For Study 1, results show that in unit 1, the *separate-RC-KC model* shows the better fit than the *integrated-RC-KC models*. By contrast, in unit 2, the *integrated-RC-KC model-1* shows better model fit than the *separate-RC-KC model* in terms of AIC and RMSE (but based on BIC the *integrated-RC-KC model-2* shows the best model fit). This supports hypothesis 2.

For Study 2, we found that in unit 1, the *separate-RC-KC model* shows a better model fit than the *integrated-RC-KC models* in terms of AIC as shown in Table 5. However, the BIC and RMSE scores indicated that the *integrated-RC-KC model-1* had the best model fit. By contrast, in unit 2, the *integrated-RC-KC model-2* shows the best model fit. This result supports hypothesis 2 in terms of AIC scores. For BIC and RMSE scores, the results partially support hypothesis 2, because the *integrated-RC-KC model-1* shows the better model fit than the *integrated-RC-KC model-2*.

Table 3. Test of hypothesis 1: Model accuracy for study 1 and study 2. Bold stands for the best fit.

Study	KC Model	# of KC	AIC	BIC	RMSE
1	<i>both-content-and-RC</i>	42	22,123	24,221	0.4422
	<i>only-RC</i>	34	22,736	24,709	0.4456
	<i>only-content</i>	9	23,853	25,446	0.4537
2	<i>both-content-and-RC</i>	42	18,559	20,358	0.4370
	<i>only-RC</i>	34	19,120	20,796	0.4421
	<i>only-content</i>	9	20,133	21,418	0.4544

Table 4. Test of hypothesis 2: Model fit by unit (Study1). Bold stands for the best fit.

Unit	RC-KC Model	# of KC	AIC	BIC	RMSE
1	<i>separate-RC-KC model</i>	33	9,733	11,488	0.4367
	<i>integrated-RC-KC model-1</i>	30	9,779	11,491	0.4368
	<i>integrated-RC-KC model-2</i>	27	9,803	11,492	0.4374
2	<i>separate-RC-KC model</i>	33	10,387	12,133	0.4414
	<i>integrated-RC-KC model-1</i>	30	10,372	12,075	0.4387
	<i>integrated-RC-KC model-2</i>	27	10,396	12,054	0.4399

Table 5. Test of hypothesis 2: Model fit by unit (Study 2). Bold stands for the best fit.

Unit	RC-KC Model	# of KC	AIC	BIC	RMSE
1	<i>separate-RC-KC model</i>	33	8,134	9,587	0.4445
	<i>integrated-RC-KC model-1</i>	30	8,156	9,567	0.4431
	<i>integrated-RC-KC model-2</i>	27	8,201	9,570	0.4461
2	<i>separate-RC-KC model</i>	33	9,095	10,598	0.4278
	<i>integrated-RC-KC model-1</i>	30	9,085	10,546	0.4263
	<i>integrated-RC-KC model-2</i>	27	9,080	10,498	0.4264

7. DISCUSSION

Our results show that the knowledge-component model incorporating both representational competencies and content knowledge had the best model fit (hypothesis 1). This result aligns with findings from a previous study that investigated whether a knowledge-component model should incorporate representational competencies using chemistry students' learning data. Our results replicate this finding in the electrical engineering domain.

Further, we found that students' representational competencies become abstracted and integrated with content knowledge as students practice representational competencies (hypothesis 2). This shows that as students' learning progress, their representational competencies are gradually merged with content knowledge. We note that the time at which each representational competency is integrated into the content knowledge may not be uniform. For instance, while students' representational competency of translating a time-domain graph to a phasor graph were merged with content knowledge in unit 2, the representational competency of translating a phasor graph to a time-domain graph remained separate from content knowledge. It is possible that after more practice, the latter representational competency would also merge with content knowledge.

Additionally, our results indirectly suggest that students' timelines may differ depending on their learning rates. Specifically, we found differences between the Study 1 and Study 2 cohorts. The fact that BIC, AIC, RMSE disagreed as to whether the *integrated-RC-KC model-1* or the *separate-RC-KC model* had a better model fit for unit 1 suggests that students in Study 2 started with somewhat more integrated knowledge and ended with more integrated knowledge compared to students in Study 1. We do not want to speculate what might have caused these cohort effects because there are numerous possible reasons, but it suggests that students may start and end at different points on a separate-to-integrated knowledge trajectory.

Our study makes novel contributions to the field of educational data mining because it is, to our knowledge, the first study capturing dynamic development of students' representational competencies using knowledge-component models. Knowledge-component modeling allowed us to identify dynamic, developmental patterns of representational competencies and to show that they are not static. Further, our finding that knowledge-component models should incorporate representational competencies in addition to

content knowledge expands the search space for knowledge-component models in future work.

Our findings also have important implications for the design of adaptive educational technologies. First, technologies that use visual representations should trace students' acquisition of representational competencies in addition to content knowledge. Doing so is particularly important at the beginning of a learning sequence. When students first learn a new concept with visual materials, instructional supports should be designed with consideration of students' representational competencies and content knowledge. Second, as students' representational competencies change dynamically through practice, the educational technology may no longer need to monitor representational competencies separately from content knowledge. This, however, may need to be adapted to the rate at which students learn specific representational competencies and content knowledge.

8. LIMITATIONS & FUTURE WORK

Our results should be interpreted considering the following limitations. First, we collected log data from students working on Signals Tutor in the context of online learning. Online learning differs from in-person learning in multiple ways. Therefore, future studies should replicate our results in the context of in-person learning. Second, our experiment was constructed in a specific electrical engineering course. Even though visual representations are commonly used in many STEM instructions, representational competencies are domain-specific and highly dependent on the particular content knowledge covered. Thus, future research needs to test whether our results generalize to other STEM domains and topics. Finally, open questions remain about the suitable length of a learning intervention to ensure that all representational competencies become integrated with content knowledge. Although this study found that the students' representational competencies are gradually integrated with content knowledge, it did not examine when each representational competency becomes fully integrated into the content knowledge. To address this limitation, a longer intervention is needed. Such research could help establish the length of learning trajectories that relate to representational competencies.

9. CONCLUSION

The present paper shows that the structure of students' knowledge changes over time. Consequently, different knowledge-component models are best suited at different times during a learning trajectory. While modeling representational competencies is important, representational competencies become integrated with content knowledge with practice. Thus, educational technologies should employ dynamic knowledge-component models that capture representational competencies separately from content knowledge at the beginning of a learning trajectory while merging them with content knowledge later in a learning sequence. The way in which these knowledge components are merged may depend on the student's learning rate. Given that prior research shows that adapting instructional support to students' representational competencies can significantly enhance their learning of content knowledge and given the prevalence of visual representations in STEM instruction, our study may have considerable impact on the effectiveness of educational technologies.

10. ACKNOWLEDGMENTS

This work was supported by NSF DUE 1933078 (IUSE). We also thank Talha Sultan, Eduardo Arvelo, and the teaching assistants for their help with this project.

11. REFERENCES

- [1] Ainsworth, S. 2006. DeFT: A conceptual framework for considering learning with multiple representations. *Learning and Instruction* 16, 3 (2006), 183–198.
- [2] Arcavi, A. 2003. The role of visual representations in the learning of mathematics. *Educational Studies in Mathematics* 52, 3 (2003), 215–241.
- [3] Cavanaugh, J.E. and Neath, A.A. 2019. The Akaike information criterion: Background, derivation, properties, application, interpretation, and refinements. *Wiley Interdisciplinary Reviews: Computational Statistics* 11, 3 (2019), e1460.
- [4] Cen, H. et al. 2006. Learning Factors Analysis – A General Method for Cognitive Model Evaluation and Improvement. *Intelligent Tutoring Systems* (Berlin, Heidelberg, 2006), 164–175.
- [5] Chi, M.T.H. et al. 1981. Categorization and representation of physics problems by experts and novices. *Cognitive Science* 5, 2 (1981), 121–152.
- [6] Corbett, A. et al. 2000. Modeling Student Knowledge: Cognitive Tutors in High School and College. *User Modeling and User-Adapted Interaction* 10, 2 (2000), 81–108.
- [7] Corbett, A.T. and Anderson, J.R. 1994. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction* 4, 4 (1994), 253–278.
- [8] Desmarais, M.C. and Naceur, R. 2013. A Matrix Factorization Method for Mapping Items to Skills and for Enhancing Expert-Based Q-Matrices. *Artificial Intelligence in Education* (Memphis, TN, USA, 2013), AIED 2013. Springer, Berlin, Heidelberg, 441–450.
- [9] diSessa, A.A. 2004. Metarepresentation: Native Competence and Targets for Instruction. *Cognition and Instruction* 22, 3 (2004), 293–331.
- [10] Dreyfus, H.L. et al. 1986. Five steps from novice to expert. *Mind over machine: The power of human intuition and expertise in the era of the computer*. (1986), 16–51.
- [11] Gentner, D. and Markman, A.B. 1997. Structure mapping in analogy and similarity. *American Psychologist* 52, 1 (1997), 45–56.
- [12] Kellman, P.J. and Garrigan, P. 2009. Perceptual learning and human expertise. *Physics of Life Reviews* 6, 2 (2009), 53–84.
- [13] Koedinger, K.R. et al. 2012. *Automated Student Model Improvement*. International Educational Data Mining Society.
- [14] Koedinger, K.R. et al. 2015. Data mining and education. *Wiley Interdisciplinary Reviews: Cognitive Science* 6, 4 (2015), 333–353.
- [15] Koedinger, K.R. et al. 2012. The Knowledge-Learning-Instruction Framework: Bridging the Science-Practice Chasm to Enhance Robust Student Learning. *Cognitive Science* 36, 5 (2012), 757–798.
- [16] Koedinger, K.R. et al. 2013. Using Data-Driven Discovery of Better Student Models to Improve Student Learning. *Artificial Intelligence in Education*, (Memphis, TN, USA, 2013), AIED 2013. Springer, Berlin, Heidelberg, 421–430.
- [17] Kozma, R. 2003. The material features of multiple representations and their cognitive and social affordances for science understanding. *Learning and Instruction* 13, 2 (2003), 205–226.
- [18] Long, Y. et al. 2018. What exactly do students learn when they practice equation solving? refining knowledge components with the additive factors model. *Proceedings of the 8th International Conference on Learning Analytics and Knowledge* (New York, NY, USA, 2018), 399–408.
- [19] Lorah, J. and Womack, A. 2019. Value of sample size for computation of the Bayesian information criterion (BIC) in multilevel modeling. *Behavior Research Methods* 51, 1 (2019), 440–450.
- [20] Nguyen, H. et al. 2019. Using Knowledge Component Modeling to Increase Domain Understanding in a Digital Learning Game. *International Conference on Educational Data Mining* (Montreal, Canada, Jul 2 - 5, 2019), 139–148.
- [21] Papousek, J. et al. 2014. Adaptive Practice of Facts in Domains with Varied Prior Knowledge. *Proceedings of the 7th International Conference on Educational Data Mining* (London, UK, July 4-7, 2014), 6–13.
- [22] Rau, M.A. 2017. A Framework for Educational Technologies that Support Representational Competencies. *IEEE Transactions on Learning Technologies* 10, 3 (2017), 290–305.
- [23] Rau, M.A. et al. 2021. Adaptive support for representational competencies during technology-based problem solving in chemistry. *Journal of the Learning Sciences* 30, 2 (2021), 163–203.
- [24] Rau, M.A. 2017. Conditions for the Effectiveness of Multiple Visual Representations in Enhancing STEM Learning. *Educational Psychology Review* 29, (2017), 717–761.
- [25] Rau, M.A. 2017. Do Knowledge-Component Models Need to Incorporate Representational Competencies? *International Journal of Artificial Intelligence in Education*. 27, 2 (. 2017), 298–319.
- [26] Rho, J. and Rau, M. A. Preparing future learning with novel visuals by supporting representational competencies. *Artificial Intelligence in Education* (Durham, UK, July 27 - 31, 2022), Springer, Cham, in press.
- [27] Ritter, S. et al. 2007. What evidence matters? A randomized field trial of Cognitive Tutor Algebra I. *Frontiers in artificial intelligence and applications* 162, (2007), 13.
- [28] Schnotz, W. 2005. An integrated model of text and picture comprehension. *The Cambridge handbook of multimedia learning* 49, (2005), 69.
- [29] Seufert, T. 2003. Supporting coherence formation in learning from multiple representations. *Learning and Instruction* 13, 2 (2003), 227–237.
- [30] VanLehn, K. 2011. The Relative Effectiveness of Human Tutoring, Intelligent Tutoring Systems, and Other Tutoring Systems. *Educational Psychologist* 46, 4 (2011), 197–221.
- [31] Villano, M. 1992. Probabilistic student models: Bayesian Belief Networks and Knowledge Space Theory. *Intelligent Tutoring Systems* (Berlin, Heidelberg, 1992), 491–498.
- [32] Khakaj, F. et al. 2016. How Teachers Use Data to Help Students Learn: Contextual Inquiry for the Design of a Dashboard. *Adaptive and Adaptable Learning* 9891, (2016), 340–354.
- [33] Yates, K.A. and Clark, R.E. 2012. Cognitive task analysis. *International Handbook of Student Achievement*. New York, Routledge. (2012).

Adversarial bandits for drawing generalizable conclusions in non-adversarial experiments: an empirical study

Yang Zhi-Han
Carleton College
yangz2@carleton.edu

Shiyue Zhang
Carleton College
zhangs@carleton.edu

Anna N. Rafferty
Carleton College
arafferty@carleton.edu

ABSTRACT

Online educational technologies facilitate pedagogical experimentation, but typical experimental designs assign a fixed proportion of students to each condition, even if early results suggest some are ineffective. Experimental designs using multi-armed bandit (MAB) algorithms vary the probability of condition assignment for a new student based on prior results, placing more students in more effective conditions. While stochastic MAB algorithms have been used for educational experiments, they collect data that decreases power and increases false positive rates [22]. Instead, we propose using adversarial MAB algorithms, which are less exploitative and thus may exhibit more robustness. Through simulations involving data from 20+ educational experiments [29], we show data collected using adversarial MAB algorithms does not have the statistical downsides of that from stochastic MAB algorithms. Further, we explore how differences in condition variability (e.g., performance gaps between students being narrowed by an intervention) impact MAB versus uniform experimental design. Data from stochastic MAB algorithms systematically reduce power when the better arm is less variable, while increasing it when the better arm is more variable; data from the adversarial MAB algorithms results in the same statistical power as uniform assignment. Overall, these results demonstrate that adversarial MAB algorithms are a viable “off-the-shelf” solution for researchers who want to preserve the statistical power of standard experimental designs while also benefiting student participants.

Keywords

MAB, bandits, experimental design, hypothesis testing

1. INTRODUCTION

Digital educational technologies offer unique opportunities to conduct pedagogical experiments and learn how to improve student outcomes. For example, experimenters can compare worked examples versus tutoring [19] or vary an avatar’s dialect [9]. When an intervention’s impact is mea-

sured soon after the intervention (e.g., via response times as in [32] or later problem correctness as in [18]), real-time data could be used to direct more students to more effective conditions. Multi-armed bandit (MAB) algorithms have been proposed as a way to conduct such adaptive experiments [15] and used for optimizing A/B comparisons (e.g. [16, 25, 26]).

While MAB assignment tends to improve outcomes for participants, it poses problems for drawing conclusions from the data. Prior research has shown systematic measurement errors, increases in false positive rate (FPR), and decreases in power when stochastic MAB algorithms are used for A/B comparisons (e.g., [22]). Potential benefits to student participants could thus be outweighed by harm to the research: lower power decreases the probability that effective interventions will be detected and deployed outside the trial, and higher FPR may lead to deploying unhelpful interventions at significant cost. While developing new algorithms and analysis approaches for these challenges is an active area of research (e.g., [5, 8]), these approaches are not yet an “off-the-shelf” solution for MAB-based experimental design.

In this paper, we consider the impact of using MAB algorithms that make performance guarantees in the adversarial case. These algorithms make weaker assumptions about their environment [4], decreasing the degree to which they can assign more students to a perceived better condition. Yet, we hypothesize that these relaxed assumptions may also decrease the *negative* consequences for drawing conclusions from the data, resulting in collected data that is more robust to the realities of educational experiments. Adversarial MAB algorithms could thus be used by researchers who want some of the benefits of condition-assignment via MAB algorithms but where their primary focus remains on drawing generalizable conclusions. These algorithms could also be *more* effective than uniform random assignment in some cases, as they can be sensitive to condition variability, and allocating more participants to an extremely variable condition can result in a better measurement of its effectiveness.

Using simulations, we first explore how 22 previously conducted experiments [29] might have been impacted if conditions had been assigned with a stochastic bandit algorithm (Thompson sampling [30]) or with one of three adversarial MAB algorithms in the Exp family [4], rather than with uniform random assignment.¹ These experiments were all conducted in ASSISTments homework assignments [11],

Y. Zhi-Han, S. Zhang, and A. Rafferty. Adversarial bandits for drawing generalizable conclusions in non-adversarial experiments: an empirical study. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 353–360, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853039>

¹All code: <http://tiny.cc/MABExpDesign> (OSF link).

leading to a thorough investigation of how these algorithms might perform in real-world settings. We go beyond prior work that has often focused on binary outcomes for students (e.g. [35]) to examine real-valued outcomes that may follow idiosyncratic distribution patterns. We show that the Exp family of adversarial bandit algorithms largely avoids the measurement and hypothesis testing inaccuracies incurred by stochastic bandits, while still providing a small but reliable improvement in average student outcomes.

We then turn to exploring MAB algorithms in the context of experiments where conditions differ in variance, which occurs when interventions narrow (or widen) gaps among students. We find that the potential power advantage for adversarial MAB algorithms is not realized in practice, although there are some scenarios in which the more exploitative stochastic MAB algorithm does increase power.

Overall, we make the following contributions: (a) introducing the idea of adversarial and stochastic-adversarial MAB algorithms as an off-the-shelf solution for allocating students to conditions; (b) demonstrating that these MAB algorithms have minimal detrimental impacts on the performance of statistical hypothesis testing in a range of real-world educational experiments; and (c) illustrating that differences in variance across two conditions are not sufficient for condition assignment with adversarial MAB algorithms to increase power over uniform assignment. These results suggest that adversarial and stochastic-adversarial MAB algorithms offer a good solution for researchers who want to improve student participant’s experiences without negatively impacting their own ability to learn from the experiment and improve experiences for the many students who are not participants.

2. RELATED WORK

Assigning participants to experimental conditions using MAB algorithms has been proposed as an alternative to uniform assignment (e.g., [12, 31, 34]). In clinical trials, a variety of methods that adapt based on previous results have been presented (e.g., [3, 5, 7], often to more quickly use data to benefit patients (e.g., [36]). More closely related to our work, applications of MAB algorithms to educational settings can help to more quickly identify pedagogically effective conditions (e.g., [15, 16, 33]) and lower barriers to teachers conducting experiments in their classrooms [35]. While MAB algorithms have also been used in education to assign students to educational interventions with the sole goal of producing the best outcomes for that particular student (e.g., [6, 13]), we focus here specifically on cases where there is a desire to extrapolate beyond individual students and draw generalizable conclusions, as in traditional scientific experiments.

While MAB algorithms are a natural choice for assigning participants to conditions to limit how many students are assigned to less effective conditions, these algorithms have consequences for researchers’ ability to use the collected data to draw conclusions about the relative effectiveness of conditions. Traditional stochastic MAB algorithms minimize regret, but this can lead to systematically biased estimates of arm means (e.g., [5, 20, 37]). They also impact traditional statistical hypothesis testing: false positive rates, where there is no actual effect in the population but the test points to there being an effect, can be increased, and

power, which measures how often a test will detect a difference when there is one, can be decreased (see e.g., [22], for specific documentation of these phenomena with Thompson sampling [1, 30], which we compare to in this paper). A variety of approaches have been taken to addressing these issues, including both statistical approaches to create unbiased estimators from the collected data (e.g., [5]) and algorithmic approaches that incorporate measurement into the algorithms’ objective (e.g., by including estimation accuracy in the objective [8] or incorporating lower bounds on power [38]). Using such approaches off-the-shelf can be challenging: the power-constrained bandits algorithm [38] makes multiple decisions about one participant, rather than learning across participants, and researchers may wish to use their standard statistical estimators and tests rather than switching to a different paradigm. In this paper, we take a slightly different approach by exploring an alternative class of MAB algorithms and examining their performance in real-world scenarios.

3. BANDIT-DRIVEN EXPERIMENTS

MAB problems are a kind of reinforcement learning problem focused on maximizing immediate rewards. A classic example is allocating limited pulls to a set of slot machines. In these problems, agents must balance collecting new information about little explored arms and exploiting information from received rewards. Here, we model selecting better educational interventions as a MAB problem: Each intervention is an arm (action) the system can choose, and initially, rewards are unknown. After a student experiences an intervention, the system receives a stochastic reward (e.g., a measure of the student’s understanding/efficiency) that influences the arm choice for the next student.

Stochastic bandit algorithms. These algorithms assume there is some stationary reward distribution underlying each arm. Here, we focus on Thompson sampling (TS; [2, 30]), which maintains an estimate of the reward distribution of each arm. From the reward that it gets from each choice, it updates this distribution. At each time step, the algorithm samples from the posterior reward distribution of each arm and chooses the arm that has the highest sampled value.

Adversarial bandit algorithms. Adversarial bandits make no statistical assumptions about the reward distribution of each arm, making them appropriate for reward distributions that are non-stationary or of unknown form. Because of the lack of assumptions, they are designed to explore more strongly and adapt more quickly to perceived changes in reward distributions. If the rewards in fact follow stationary distributions, this can lead to lower expected reward than stochastic bandit algorithms, but when reward distributions deviate from these assumptions, adversarial bandit algorithms have stronger performance guarantees than stochastic bandit algorithms. From the perspective of using MAB algorithms for experimental design, the extra exploration in adversarial algorithms could collect better data for drawing conclusions about differences between arms, albeit while lowering benefits for participants in the experiment.

In this work, we focus on the popular Exp3 family of adversarial bandit algorithms [4]. Exp3 balances exploration and exploitation via an exploration hyperparameter that influ-

ences both the probability of picking an arm uniformly at random and the strength of response to high rewards from low-probability arms. This hyperparameter allows an experimenter to adjust the amount of exploration, potentially increasing reward at the cost of collecting more biased data. Because this choice is difficult to optimize ahead of time, we also examine the performance of Exp3.1, which eliminates the hyperparameter and provides worst case performance guarantees regardless of the true reward distributions [4].

One algorithm that has performance guarantees in both stochastic and adversarial environments is Exp3++ [28]. While this algorithm achieves lower expected rewards than TS, it often improves upon the obtained reward of Exp3 while still employing enough exploration to perform well in adversarial environments. It also can be used off-the-shelf, with fixed values for the hyperparameters that probabilistically guarantee asymptotic performance [27].

We want to explore how well adversarial and stochastic MAB algorithms meet the needs of researchers for data collection in educational experiments and how they impact student experiences compared to traditional uniform assignment. We hypothesize that the adversarial bandit algorithms (Exp3 with a fixed value of 0.05 for the hyperparameter, Exp3.1, Exp3++) will have comparable performance for collecting research data to uniform random allocation, with benefits to students that are greater than uniform random allocation but less than those from a representative stochastic bandit algorithm (TS). Further, Exp3++ is likely to improve on the purely adversarial algorithms’ performances in assigning more students to better arms.

4. EVALUATING ADVERSARIAL BANDITS: ASSISTMENTS EXPERIMENTS

The probabilistic asymptotic performance guarantees of MAB algorithms, both in stochastic and adversarial environments, suggest that student participants would benefit if these algorithms were used for experimental design. However, these guarantees do not speak to how biased the collected data will be, nor whether standard statistical hypothesis testing will be able to draw accurate conclusions from that data. Further, real educational experiments may have non-normally distributed outcome measures, impacting the collected data and performance of each algorithm. To explore how well the Exp3 variants meet the needs of researchers to draw accurate conclusions and the desire to place more students in a better condition, we conduct simulations that leverage previously collected datasets from educational experiments, conducted on the ASSISTMENTS platform [11]. We use these datasets as a case study for how to apply MAB algorithms to scenarios with varying, real-valued reward distributions, where some students reach mastery quickly and others never do so, and by repeatedly simulating the potential impact of each of the four bandit algorithms (TS, Exp3, Exp3.1, Exp3++), we can measure statistical power, false positive rate (FPR), and accuracy of arm measurement across algorithms.

4.1 Methods

4.1.1 Modeling real-world datasets

We focus on datasets from 22 randomized controlled experiments run inside the *SkillBuilder* interface of the ASSIST-

ments online learning platform [29], which focuses primarily on 4th-12th grade math. These datasets included a total of 14,947 students in grades 5-12 (25% of students lacked a reported grade). Students had 200 different teachers and were drawn from 19 states (states deidentified in the data), and included a “guessed” gender based on name for 68% of students (of these, 53% were female; see [10] for information on gender methodology). No information on student race/ethnicity or SES was available, and experiments were IRB approved; see [29] for more dataset details.

In each experiment, students were placed into one of two conditions when completing homework. Each student must answer several *consecutive* problems correctly to complete the homework (typically three), and the number of problems P the student attempted before completion was recorded. Both completing homework and doing so in fewer problems are desirable, and we translate these measures into a reward signal for the MAB algorithms. To eliminate scaling issues, rewards are scaled to a fixed range as follows. Based on examination of the range of problems to completion across experiments, we cap the maximum number of problems at 30. If $P \geq 30$ or the student did not complete the homework, then we set P to 30. Because lower values of P are better, reward is then $r = 30 - P$. This reward is guaranteed to be in $[0, 30]$ and is then linearly interpolated into $[0, 1]$ for the MAB algorithms. We refer to the better condition as arm 1 and the worse condition as arm 2 on all datasets.

To conduct repeated trials with data from previously conducted experiments, our framework *resamples* an outcome associated with the chosen condition in the dataset when that condition is assigned to an incoming student. Within each trial, we fix the number of students n to the number in the original experiment ($n \in [129, 1797]$).

4.1.2 Simulation setup

To assess the ability of traditional hypothesis tests to draw accurate conclusions from the collected data, we measure (1) power – the proportion of the time an effect is detected when one exists – using scenarios where two conditions are *different* and (2) false positive rate (FPR) – the proportion of the time an effect is falsely detected when one does not exist – using scenarios where two conditions are the *same* in terms of expected reward. For (1), we focus on the seven ASSISTMENTS datasets with the largest effect sizes measured in terms of Cohen’s d ($0.16 \sim 0.51$), as larger effect sizes are more likely to reflect educationally relevant differences and may lead to larger differences across algorithms; we refer to these as ASSISTMENTS-GES (greater effect size). We examine how the allocation methods impact average reward, which measures student outcomes, and statistical power and measured arm means, which are of large importance to researchers. For (2), we create modified datasets in which each condition’s reward list contains all rewards from *both* arms. This creates two conditions with the same expected outcome while keeping their realistic reward distributions. We refer to these 22 modified datasets as ASSISTMENTS-RC (reward combined). Because no allocation method could increase benefits to students given that the conditions do not differ, we focus here on examining FPR and measured arm means. For both (1) and (2), we follow Section 4.1.1 and run 1000 trials for each dataset-algorithm combination.

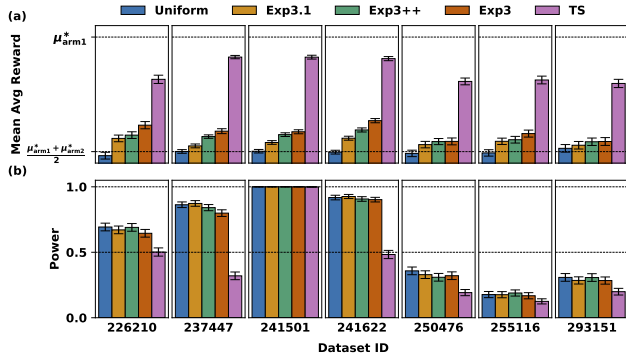


Figure 1: In terms of benefiting students as measured by mean average reward (a), all Exp allocations under-performed TS but out-performed uniform (except on dataset 293151). This was possible even when Exp3.1 and Exp3++ allocation achieved comparable power to uniform allocation, as indicated by overlapping error bars on all seven datasets (b). Error bars show $\pm 1.96 \times \text{SE}$.

4.1.3 Data collection and analysis

For each trial, we record all condition assignments and unscaled rewards. We then compute: the average reward per student, the average reward $\hat{\mu}$ for each condition (true expected reward denoted by μ^*), and the conclusion of a two-sided hypothesis test of whether the two conditions differ at a population level. Because capping P as discussed in Section 4.1.1 can lead to strong bimodality in reward distributions, which dissatisfies the assumptions of a standard t -test, we use the non-parametric Brunner-Munzel test for testing for a difference between conditions in the collected data. This test assumes neither normality nor equivariance of the distributions from which the samples are drawn. We consider the test to detect an effect if and only if $p < .05$.

To determine if a statistic differed reliably based on condition allocation method, we use generalized linear regression, with factors for algorithm (with uniform as reference group) and ASSISTments dataset. We report two decimal places except for small or similar values.

4.2 Results

4.2.1 Datasets with effects: ASSISTments-GES

Mean average reward: All MAB allocations were associated with significantly higher benefits to students, as measured by mean average reward, than uniform allocation (coefficient for TS: 0.80; Exp3: 0.19; Exp3.1: 0.10; Exp3++: 0.15; all $p < .001$). TS collected data with highest mean average reward (16.34), followed by Exp3 (15.74; 24% of the reward gain of TS over uniform), Exp3++ (15.69; 19% of TS over uniform), Exp3.1 (15.64; 12% of TS over uniform), and Uniform (15.54). For a breakdown by dataset, see Figure 1a.

Power: TS and Exp3 allocations had significantly lower power than uniform (coefficient for TS: -1.42 ; Exp3: -0.18 ; both $p < .001$), while allocation with the other two algorithms did not (Exp3.1: -0.05 , $p = .22$; Exp3++: -0.07 , $p = .11$). On average, TS collected data with lowest power (0.40), followed by Exp3 (0.59), Exp3++ (0.61), Exp3.1 (0.61) and

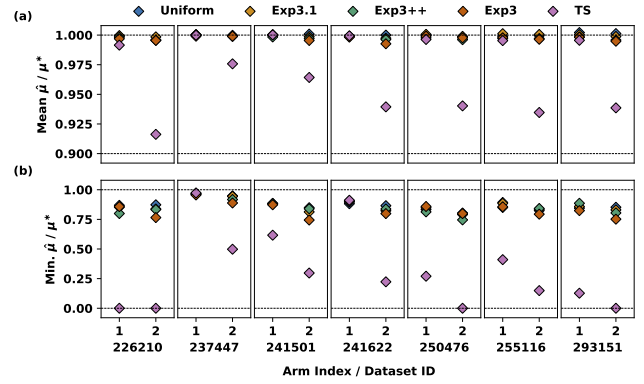


Figure 2: TS allocation led to a clear, systematic underestimation of arm 2 [worse arm] on all seven datasets (a), while Exp allocations resulted in arm mean estimates similar to those of Uniform (a). Error bars are not shown for clarity and are delegated to hypothesis testing. Also, the worst-case $\hat{\mu}$'s are much worse for TS allocation than for Exp (b).

Uniform (0.62). Figure 1b shows a breakdown by dataset.

Measured arm means: Our work replicates prior work showing that TS underestimates the worse arm in binary trials [22] and further finds a small underestimation of the better arm (arm 1 [better arm] coefficient: -0.05 ; arm 2 [worse arm] coefficient: -0.76 ; all $p < .001$). The Exp algorithms resulted in more accurate measurement of arm means: while Exp3 and Exp3++ also underestimate both arms, the extent to which they underestimate the worse arm is much less than TS (for Exp3: arm 1 coefficient: -0.02 , $p < .05$; arm 2 coefficient: -0.04 , $p < .05$; for Exp3++: arm 1 coefficient: -0.02 , $p < .01$; arm 2 coefficient: -0.04 , $p < .05$). Arm mean estimates using data from Exp3.1 did not differ significantly from those derived from uniform allocation (arm 1 coefficient: 0.004 , $p = .71$; arm 2 coefficient: -0.007 , $p = .69$). See Figure 2a for a qualitative comparison by dataset and Figure 2b for a worst-case analysis showing results for the trial with the most inaccurate $\hat{\mu}$.

4.2.2 Datasets without effects: ASSISTments-RC

FPR: TS allocation was associated with significantly higher FPR than uniform allocation (coefficient for TS: 0.56 , $p < .001$), while allocation with the other three algorithms was not (coefficient for Exp3: 0.0065 , $p = .88$; for Exp3.1: -0.0019 , $p = .97$; for Exp3++: -0.0047 , $p = .91$). On average, TS collected data with the highest FPR (0.0867) and is followed by Exp3 (0.0517), Uniform (0.0514), Exp3.1 (0.0513) and Exp3++ (0.0512). As shown in Figure 3a, TS inflates FPR for almost all datasets, while the other algorithms have FPR $< .06$ for the vast majority of datasets.

Measured arm means: Since the two arms are identical and within a trial, their average values are not independent, we arbitrarily examine one of the two arms for each trial. Allocation using TS, Exp3, and Exp3++ was associated with significantly lower estimates of arm means compared to uniform allocation (coefficient for TS: -0.42 ; Exp3: -0.0320 ; Exp3++: -0.0274 ; all $p < .001$); note that the bias for TS is much larger than for Exp3 and Exp3++. Similar results

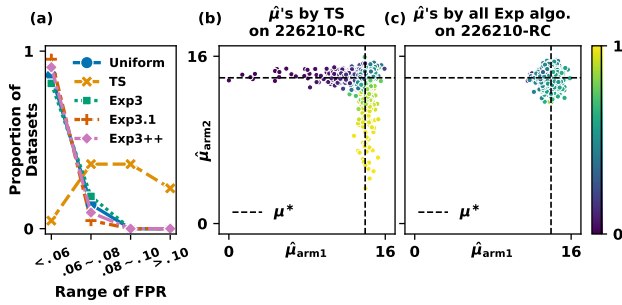


Figure 3: (a) TS allocation results in more datasets with inflated FPR than uniform and adversarial allocations. Remaining figures show the distribution of $(\hat{\mu}_{\text{arm1}}, \hat{\mu}_{\text{arm2}})$ pairs of TS (b) and of all Exp algorithms (c) on dataset 226210-RC, with hue showing $N_{\text{arm1}}/N_{\text{total}}$. Clearly, TS has trials where one arm is very badly estimated while Exp algorithms don't. Hue offers an explanation: for TS (b), $\hat{\mu}_{\text{arm1}}$ is roughly unbiased when arm 1 is pulled more often (a proxy of appearing to be better) and negatively biased when it is pulled less often (a proxy of appearing to be worse early-on); Exp algorithms (c) allocate more evenly, indicating that it explored both arms beyond initial under/over-estimation.

hold if we arbitrarily analyze the other arm instead. Exp3.1 did not have a significant negative bias with one choice of arm for analysis (coefficient: -0.0092 , $p = .23$), but did for the other (coefficient: -0.0173 , $p < .05$), suggesting a weak negative bias in estimated mean. This replicates results for TS from prior work [22] and shows the same underestimation can occur for adversarial bandits, albeit with much smaller magnitude of underestimation: in all cases, the MAB algorithm underestimates the mean of the arm that appears, due to random sampling, to be worse, and then samples the arm that appears to be better more often, leading the estimate of the other arm to be below its true mean. See Figure 3b and c for a comparison between TS and Exp algorithms.

Overall, these results show that experimental design using the Exp family of algorithms can collect data that meets researchers' needs better than a purely stochastic bandit algorithm, with higher power, lower FPRs, and more accurate condition measurement. These algorithms do not benefit students as much as the purely stochastic TS, but all Exp algorithms do improve on uniform allocation. Exp3++ provides a balance between Exp3.1 and Exp3, achieving 19% of the reward gain of TS, requiring no hyper-parameters, and demonstrating only a slight underestimation of arm means.

5. POWER AND UNEQUAL VARIANCE

The ASSISTments simulations demonstrate that across a range of educational experiments, the Exp algorithms performed better when measured in terms of researchers' concerns, like power and arm mean estimates, while TS attained greater benefits for students. More generally, both TS and Exp aim to optimize reward, but they do so with different assumptions about the environment, meaning that the specific characteristics of an experiment will influence how well each performs on both student- and researcher-centric measures. However, because the ASSISTments experiments do

not vary systematically from one another, they are not an ideal platform for exploring the impact of *specific* experimental characteristics on the MAB algorithms' performance compared to one another and compared to uniform allocation. We thus turn to simulations with constructed datasets to explore the impact of one experimental characteristic: the relative variability of the two conditions.

Ideally, pedagogical interventions increase equity and narrow achievement gaps between students, indicated by lower variability among students who experience the intervention, but in the non-ideal case they also could widen these gaps. Interestingly, differences in condition variability affect what allocation of students is best for power: uneven allocation that places more students in the more variable condition will result in higher power than uniform allocation, at the potential cost of reward. In the simulations that follow, we examine how allocation using TS and Exp algorithms juggles power with reward differently, as well as whether Exp algorithms' adversarial assumptions and sensitivity to condition variability make them improve upon uniform allocation for measures like power that are researchers' primary concern.

5.1 Methods

5.1.1 Two-arm scenarios

To systematically investigate the impact of having conditions that differ in variability, we construct artificial scenarios with two arms that have normally distributed rewards. We fix the expected reward of arm 1 as 1 and that of arm 2 as 0, and vary whether the better or worse arm has a higher variance and the magnitude of the differences in variance; specifically we consider the following 19 scenarios:

$$\{(1, 1)\} \cup \underbrace{\{\{1\} \times \{2 : 10\}\}}_{\text{Worse arm has higher SD}} \cup \underbrace{\{\{2 : 10\} \times \{1\}\}}_{\text{Better arm has higher SD}}$$

where each tuple gives the standard deviation (SD) of arm 1 followed by arm 2.

As in Section 4, rewards are interpolated into $[0, 1]$ for the MAB algorithms. Rewards are first clipped to $[-30, 31]$, where -30 is the mean of the worse arm (0) minus three times the maximum possible SD (10) and 31 is the mean of the better one (1) plus three times the maximum possible SD. We run 10000 trials for each scenario-algorithm combination, and each trial includes 250 simulated students.

5.1.2 Data collection and analysis

Data collection is described in Section 4.1.3. For analysis, we use Welch's test (t -test that does not assume equal variances) instead of the Brunner-Munzel test, as raw rewards are normally distributed and the clipping range is wide.

To determine if a statistic for data collected using the MAB algorithms differed reliably from that for data collected using uniform allocation, we use generalized linear regression, with factors for algorithm (with uniform as reference group), standard deviation of the arm with variable variance, and the interaction between the two.

5.2 Results

As shown in Figure 4, the impact of allocation method on power differed systematically based on whether the better

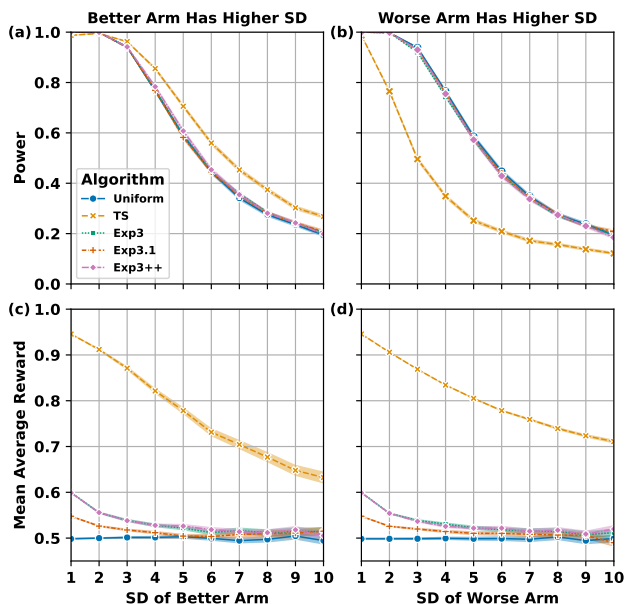


Figure 4: TS has higher power than uniform when the better arm is more variable (a), but lower power when the worse arm is more variable (b); Exp algorithms and uniform perform similarly. In all cases, MABs outperform uniform, with TS attaining the highest reward (c-d). Error bands: ± 1.96 SE.

or worse arm had higher variance. TS allocation was associated with significantly higher power than uniform allocation when the better arm was more variable (coefficient: 0.20; $p < .001$; Figure 4a), but significantly lower power when the worse arm was more variable (coefficient: -1.94 , $p < .001$; Figure 4b). In contrast, the three Exp algorithms performed qualitatively similarly to uniform allocation in both scenarios, with a small but reliable decrease in power for Exp3 and Exp3.1 when the worse arm was more variable (coefficient for Exp3: -0.13 , $p < .001$; Exp3.1: -0.09 , $p < .01$). No other power differences were detected. While the potential of TS to increase power in some situations seems promising, researchers will not know in advance whether to expect TS to increase or decrease power, and the decreases in power from TS when the worse arm has higher variance are larger than the increases in the opposite case. Further, designing an intervention where the better arm is more variable is generally undesirable: it corresponds to a scenario in which an intervention that helps students on average also widens gaps among individual students.

As expected and shown in Figure 4c-d, TS had larger reward compared than the Exp algorithms, but all MAB algorithms resulted in reliably higher reward – i.e., benefits to students – than uniform allocation (better arm more variable: coefficient for TS: 0.47; Exp3: 0.07; Exp3.1: 0.03; Exp3++: 0.07; worse arm more variable: TS: 0.45; Exp3: 0.08; Exp3.1: 0.04; Exp3++: 0.07; all $p < .001$).

6. DISCUSSION

Pedagogical experiments are a useful tool for improving education, but allocating students to conditions uniformly at random can pose challenges, with larger subject pools lead-

ing to more students experiencing an inferior educational condition but smaller subject pools potentially decreasing the ability of researchers to differentiate conditions with certainty. Our results suggest that adversarial bandit algorithms offer a way to increase the proportion of students assigned to a better condition with limited compromises to the conclusions that can be drawn from the experimental results. Hyper-parameter free adversarial bandit algorithms like Exp3++ thus offer a researcher-friendly option for experimental design that performs well even with non-standard outcome distributions, as we saw in the ASSISTments simulations. While TS improved power when the better condition had higher variance, it also decreased power when the worse arm had higher variance. Researchers are unlikely to know which of these scenarios applies before collecting results, and this unpredictability of the impact of TS, coupled with the higher FPR and lower power on average in the real educational datasets, may make stochastic bandits less attractive to researchers.

One limitation of this work is its use of simulations rather than new experiments. Importantly, simulations are needed to measure power and FPR, but field testing with adversarial algorithms is also needed to assess their real-time feasibility as well as the existence and impact of temporal trends in outcomes. All studied algorithms run in real-time on a single CPU, but students may complete homework contemporaneously, with some assigned a condition prior to another student finishing and thus without incorporating the outcome of that student. This could be handled by batch updating [17,21], but there has been limited exploration of the impact of batching on analyzing the collected data. Temporal trends may occur in experiments if, say, higher prior knowledge students complete homework first, or multiple schools participate in an experiment at different times. Adversarial bandit algorithms should outperform stochastic ones in these situations, but empirical study is necessary.

A second limitation is our assumption that one arm is better for all students, without regard for personalization. While personalization is an exciting area for future work, incorporating personalization in bandit algorithms for education poses its own ethical conundrums, including potentially lesser outcomes for less well-represented groups (see [14]).

Future work should also examine other scenarios where bandit algorithms might increase power. Here, we focused only on conditions with differing variance, but other interesting scenarios include experiments with more conditions, with predicted outcome distributions that exhibit particular non-standard characteristics (e.g., bimodality), or with different analysis goals than detecting if two conditions differ in mean. One or more types of bandit algorithms (e.g., stochastic, adversarial, or best-arm identification [23]) may be best for each scenario; based on their weak environmental assumptions, we believe adversarial bandits may be reasonable in all of these scenarios. Work in optimal experiment design (OED; see, e.g., [24]) shows the potential of non-uniform allocation to increase the information gained from an experiment. Bandit algorithms often require less setup and a priori knowledge than OED, and thus identifying information gain benefits of these algorithms in particular settings could benefit both researchers and student participants.

7. REFERENCES

- [1] S. Agrawal and N. Goyal. Analysis of Thompson sampling for the multi-armed bandit problem. In S. Mannor, N. Srebro, and R. C. Williamson, editors, *Proceedings of the 25th Annual Conference on Learning Theory*, volume 23, pages 39.1–39.26, Edinburgh, Scotland, 2012. PMLR.
- [2] S. Agrawal and N. Goyal. Thompson sampling for contextual bandits with linear payoffs. In S. Dasgupta and D. McAllester, editors, *Proceedings of the 30th International Conference on International Conference on Machine Learning*, volume 28, pages 127–135. JMLR, 2013.
- [3] A. C. Atkinson. Selecting a biased-coin design. *Statistical Science*, 29(1):144–163, 2014.
- [4] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM journal on computing*, 32(1):48–77, 2002.
- [5] J. Bowden and L. Trippa. Unbiased estimation for response adaptive clinical trials. *Statistical Methods in Medical Research*, 26(5):2376–2388, 2017.
- [6] B. Clement, D. Roy, P.-Y. Oudeyer, and M. Lopes. Multi-armed bandits for intelligent tutoring systems. *Journal of Educational Data Mining*, 7:20–48, 2015.
- [7] L. Duan and F. Hu. Doubly adaptive biased coin designs with heterogeneous responses. *Journal of Statistical Planning and Inference*, 139(9):3220–3230, 2009.
- [8] A. Erraqabi, A. Lazaric, M. Valko, E. Brunskill, and Y.-E. Liu. Trading off rewards and errors in multi-armed bandits. In A. Singh and J. Zhu, editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54, pages 709–717. PMLR, 2017.
- [9] S. Finkelstein, E. Yarzebinski, C. Vaughn, A. Ogan, and J. Cassell. The effects of culturally congruent educational technologies on student achievement. In *International Conference on Artificial Intelligence in Education*, pages 493–502. Springer, 2013.
- [10] N. T. Heffernan. Assistments data: gender. <https://sites.google.com/site/assistmentsdata/an-explanation-on-how-to-interpret-our-data-sets/gender>, 2014. [Online; accessed 21-February-2022].
- [11] N. T. Heffernan and C. L. Heffernan. The assistments ecosystem: Building a platform that brings scientists and teachers together for minimally invasive research on human learning and teaching. *International Journal of Artificial Intelligence in Education*, 24(4):470–497, 2014.
- [12] C. Kaibel and T. Biemann. Rethinking the gold standard with multi-armed bandits: Machine learning allocation algorithms for experiments. *Organizational Research Methods*, 24(1):78–103, 2021.
- [13] A. S. Lan and R. G. Baraniuk. A contextual bandits framework for personalized learning action selection. In T. Barnes, M. Chi, and M. Feng, editors, *Proceedings of the Ninth International Conference on Educational Data Mining*, pages 424–429, 2016.
- [14] Z. Li, L. Yee, N. Sauerberg, I. Sakson, J. J. Williams, and A. N. Rafferty. Getting too personal(ized): The importance of feature choice in online adaptive algorithms. In *Proceedings of the 13th International Conference on Educational Data Mining, EDM 2020, Fully virtual conference, July 10-13, 2020*, pages 159–170. International Educational Data Mining Society, 2020.
- [15] Y.-E. Liu, T. Mandel, E. Brunskill, and Z. Popovic. Trading off scientific knowledge and user learning with multi-armed bandits. In J. Stamper, Z. Pardos, M. Mavrikis, and B. McLaren, editors, *Proceedings of the 7th International Conference on Educational Data Mining*, pages 161–168, 2014.
- [16] J. D. Lomas, J. Forlizzi, N. Poonwala, N. Patel, S. Shodhan, K. Patel, K. Koedinger, and E. Brunskill. Interface design optimization as a multi-armed bandit problem. In *Proceedings of the 2016 CHI conference on human factors in computing systems*, pages 4142–4153, 2016.
- [17] T. Mandel, Y.-E. Liu, E. Brunskill, and Z. Popović. The queue method: Handling delay, heuristics, prior data, and evaluation in bandits. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 2849–2856, 2015.
- [18] P. McGuire, S. Tu, M. E. Logue, C. A. Mason, and K. Ostrow. Counterintuitive effects of online feedback in middle school math: results from a randomized controlled trial in ASSISTments. *Educational Media International*, 54(3):231–244, 2017.
- [19] B. M. McLaren, T. van Gog, C. Ganoë, M. Karabinos, and D. Yaron. The efficiency of worked examples compared to erroneous examples, tutored problem solving, and problem solving in computer-based learning environments. *Computers in Human Behavior*, 55:87–99, 2016.
- [20] X. Nie, X. Tian, J. Taylor, and J. Zou. Why adaptively collected data have negative bias and how to correct for it. In *International Conference on Artificial Intelligence and Statistics*, pages 1261–1269, 2018.
- [21] D. Provodin, P. Gajane, M. Pechenizkiy, and M. Kaptein. The impact of batch learning in stochastic bandits. *NeurIPS 2021 Workshop on Ecological Theory of Reinforcement Learning*, 2021.
- [22] A. N. Rafferty, H. Ying, and J. J. Williams. Statistical consequences of using multi-armed bandits to conduct adaptive educational experiments. *Journal of Educational Data Mining*, 11(1):47–79, 2019.
- [23] D. Russo. Simple bayesian algorithms for best-arm identification. *Operations Research*, 68(6):1625–1647, 2020.
- [24] E. G. Ryan, C. C. Drovandi, J. M. McGree, and A. N. Pettitt. A review of modern computational algorithms for bayesian optimal design. *International Statistical Review*, 84(1):128–154, 2016.
- [25] E. M. Schwartz, E. T. Bradlow, and P. S. Fader. Customer acquisition via display advertising using multi-armed bandit experiments. *Marketing Science*, 36(4):500–522, 2017.
- [26] S. L. Scott. Multi-armed bandit experiments in the online service economy. *Applied Stochastic Models in Business and Industry*, 31(1):37–45, 2015.
- [27] Y. Seldin and G. Lugosi. An improved parametrization and analysis of the exp3++ algorithm

- for stochastic and adversarial bandits. In *Conference on Learning Theory*, pages 1743–1759. PMLR, 2017.
- [28] Y. Seldin and A. Slivkins. One practical algorithm for both stochastic and adversarial bandits. In *International Conference on Machine Learning*, pages 1287–1295. PMLR, 2014.
- [29] D. Selent, T. Patikorn, and N. Heffernan. ASSISTments dataset from multiple randomized controlled experiments. In *Proceedings of the Third ACM Conference on Learning at Scale*, pages 181–184. ACM, 2016.
- [30] W. R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933.
- [31] S. S. Villar, J. Bowden, and J. Wason. Multi-armed bandit models for the optimal design of clinical trials: benefits and challenges. *Statistical science: A review journal of the Institute of Mathematical Statistics*, 30(2):199–215, 2015.
- [32] C. Walkington, V. Clinton, and A. Sparks. The effect of language modification of mathematics story problems on problem-solving in online homework. *Instructional Science*, 47(5):499–529, 2019.
- [33] J. J. Williams, J. Kim, A. Rafferty, S. Maldonado, K. Z. Gajos, W. S. Lasecki, and N. Heffernan. Axis: Generating explanations at scale with learnersourcing and machine learning. In *Proceedings of the Third ACM Conference on Learning at Scale*, pages 379–388. ACM, 2016.
- [34] J. J. Williams, A. N. Rafferty, A. Ang, D. Tingley, W. S. Lasecki, and J. Kim. Connecting instructors and learning scientists via collaborative dynamic experimentation. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, pages 3012–3018. ACM, 2017.
- [35] J. J. Williams, A. N. Rafferty, D. Tingley, A. Ang, W. S. Lasecki, and J. Kim. Enhancing online problems through instructor-centered tools for randomized experiments. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pages 207:1–207:12. ACM, 2018.
- [36] S. F. Williamson, P. Jacko, S. S. Villar, and T. Jaki. A bayesian adaptive design for clinical trials in rare diseases. *Computational statistics & data analysis*, 113:136–153, 2017.
- [37] M. Xu, T. Qin, and T.-Y. Liu. Estimation bias in multi-armed bandit algorithms for search advertising. In C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26, pages 2400–2408. Curran Associates, Inc., 2013.
- [38] J. Yao, E. Brunskill, W. Pan, S. Murphy, and F. Doshi-Velez. Power-constrained bandits. *arXiv preprint arXiv:2004.06230*, 2020.

Towards Real Interpretability of Student Success Prediction Combining Methods of XAI and Social Science

Lea Cohausz
University of Mannheim
lea@informatik.uni-mannheim.de

ABSTRACT

Despite calls to increase the focus on explainability and interpretability in EDM and, in particular, student success prediction, so that it becomes useful for personalized intervention systems, only few efforts have been undertaken in that direction so far. In this paper, we argue that this is mainly due to the limitations of current Explainable Artificial Intelligence (XAI) approaches regarding interpretability. We further argue that the issue, thus, calls for a combination of AI and social science methods utilizing the strengths of both. For this, we introduce a step-wise model of interpretability where the first step constitutes of knowing important features, the second step of understanding counterfactuals regarding a particular person's prediction, and the third step of uncovering causal relations relevant for a set of similar students. We show that LIME, a current XAI method, reaches the first but not subsequent steps. To reach step two, we propose an extension to LIME, Minimal Counterfactual-LIME, finding the smallest number of changes necessary to change a prediction. Reaching step three, however, is more involved and additionally requires theoretical and causal reasoning - to this end, we construct an easily applicable framework. Using artificial data, we showcase that our methods can recover connections among features; additionally, we demonstrate its applicability on real-life data. Limitations of our methods are discussed and collaborations with social scientists encouraged.

Keywords

Educational Data Mining, Student Drop-Out Prediction, XAI, Explainability

1. INTRODUCTION

Educational Data Mining (EDM) and in particular its sub-field of student success and dropout prediction has gained prominence in recent years due to the increased digital education data availability and because the prediction of students' successes and struggles poses an important real-life

problem. Accordingly, a multitude of studies exist testing various Machine and Deep Learning techniques on different data with some achieving remarkable accuracy and F1-values of more than 80 or even 90% regarding drop-out or success prediction [19, 13, 14, 6, 11]. This seems impressive and Xing & Du [17] write that individual drop-out probabilities can be used to "provide stronger and prioritized intervention to these students as a way of personalization" (p. 558). However, simply using the predictions will not enable one to do that. These only allow us to know *who* is likely to drop out; but in order to do anything with this prediction, we ought to know *why* the prediction has been made. Understanding why a prediction is made, is the topic of another timely topic in computer science, Explainable Artificial Intelligence (XAI). There, the *why* is split up further in *global* and *local* feature importance. With global, we mean what features are generally considered important by the model for predictions. This is valuable information so that we can control that features which would lead to a biased system discriminating against certain populations or features mistakenly included do not have an impact. A local explainability of a prediction, in contrast, relates to the importance of features regarding a specific person's prediction. This is important when we aim to use our predictions to help and advice a student predicted to be at risk as it allows us to understand what features contribute to their prediction specifically. Providing a basis to construct a personalized intervention system is the aim of this paper - thus, we work with local explainability.

The importance of XAI in EDM seems obvious and we are not the first to think so. Chitti et al. [5] heavily advocate a use of XAI techniques in future studies lamenting a lack thereof in current research. Alhamri & Alharbi [2] investigated the use of XAI and explainability techniques in performance prediction and came to the conclusion that few studies focus on it, and that those doing so merely focus on global explainability and do not employ techniques offered by XAI, instead simply using standard techniques as Decision Trees and looking at the generated rules. Indeed, it seems that with one notable exception [3], hardly any current research includes local explainability.

At first, it seems surprising that XAI techniques have not been employed more frequently as comparatively easy to use and model agnostic methods to extract important features exist¹. However, we will see in this paper, that employing

¹SHAP for global, and LIME for local explainability [10,

L. Cohausz. Towards real interpretability of student success prediction combining methods of XAI and social science. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 361–367, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853069>

XAI techniques off-the-shelf still leads to rather empty explanations. While knowing the important features explains why a prediction has been made, we also need to be able to interpret why and how these features matter at all. To better distinguish between those concepts, we argue for a distinction between *explainability* - which in this paper will relate to explaining the decision of the model, i.e. knowing the important features - and *interpretability* - which in this paper will relate to being able to interpret why and how features are important. Current XAI methods, as we will demonstrate later on, do not allow for interpretability.

Upon showing this, the objective of this paper is to provide a way of reaching interpretability as a basis for constructing personalized interventions. To this end, we will turn to another discipline, social science, that is traditionally well-equipped to deal with causal mechanisms and combine the predictive abilities of Machine Learning with the theory and causal analysis tools of social science. More precisely, we will:

- after a brief introduction to LIME, introduce a step-wise model of interpretability that will be illustrated with an example showing that XAI alone only achieves explainability but not interpretability.
- argue for a combination of XAI with social science, allowing us to gain full interpretability to construct a personalized intervention system. To this end, we will introduce a pipeline employing techniques from both disciplines.
- evaluate the approach on artificial data, so that we can test whether causal relationships are recovered; and show the applicability of the approach on real-life data.
- discuss the limitations of the approach and call for more collaborations among social scientists and computer scientists.

2. LIME

The aim of a personalized intervention system is both to provide a student with an idea of why they are predicted to struggle and what can be done to change this; and to provide pointers to those concerned with tailoring programs to help students. The requirements are, thus, to know what exact features contribute to a specific student's prediction (local explainability), which features' values would need to change in what way to change the prediction, and what underlying causal mechanisms are at work.² Note that when we consider causal mechanisms, we stray away from what

15].

²Note that while with global XAI, we often strive to exclude features that could be discriminating (e.g. age, gender, ...), local interpretability and the task of providing personalized intervention systems allows us to use those features as (a) people are not selected into programs because of it and more importantly (b) because we can try to mitigate difficulties certain populations have. If we identify, e.g., that some older students struggle, we could think about the mechanisms behind this, (e.g. older students generally have other responsibilities such as jobs and family competing for time), and and try to find solutions (e.g. provide flexible timetables, childcare on campus, ...).

XAI and purely data-driven approaches can provide. XAI techniques are concerned with explaining a model's decision but when we are interested in the causal mechanisms behind the important features we a) can no longer use XAI off-the-shelf and b) simultaneously make the assumption that the features important for the model also carry importance in real-life. The latter assumption should be kept in mind as it is not necessarily a given. Nonetheless, XAI can serve as a valuable basis for interpretability. Due to our focus on local interpretability, we choose to employ LIME. LIME is an acronym for Local Interpretable Model Agnostic Explanations which - as the name says - works for every model and finds local explanations for each instance. The basic idea is to randomly sample n feature vectors around the instance we want to explain given the normal distribution and to then weigh these new instances according to the distance to the instance we want to explain. Furthermore, the predicted label for each sample is obtained by feeding the feature vector into the model (note, that this works regardless of the specific method making it model agnostic). Based on this, Lasso regression is employed on the generated data with the predicted labels being the dependent (or target) variable. This allows us to extract the k most important features for the prediction [15].

3. A MODEL OF INTERPRETABILITY

One of the reasons why XAI techniques have not been used much may relate to their limitations regarding actual interpretability [9, 8, 1]. To illustrate our argument, consider Figure 1. It shows a Directed Acyclical Graph (DAG) of factors influencing whether a course, C1, is completed. We can observe whether courses C2, C3, and C4 are taken in parallel or not. These are our observable variables that could be used as features in a ML model. Taking these classes in parallel does not influence the completion of C1 directly. However, they do so indirectly through latent factors we cannot observe. Courses C2 and C3 compliment the contents of C1 well and taking them in parallel increases the competences required to complete C1 which then increases the probability of finishing the course. C4 is not related to C1 regarding content and thus does not contribute to competence important to complete C1. All three classes C2, C3, and C4 contribute to the workload, though. Having a high workload decreases the probability of finishing C1. Imagine now that for a student, Alice, the drop-out probability for C1 is predicted to be high. In order to fully leverage on this prediction, we should walk through each of the three steps of interpretability.

1. *Understand which features matter.* This means that we know which features matter for a person's prediction (local explainability). In our example, this means that the most important features regarding Alice's prediction are revealed to be the parallel taking of C2, C3, and C4. Furthermore, we know at this step that all three have a negative impact on completing C1. This is good to know but knowing the direction and impact of features is not equal to knowing what has to change in order to change the prediction. Should Alice not take any of these classes in parallel?
2. *Understand what would need to change to change the prediction.* In other words, we are looking for coun-

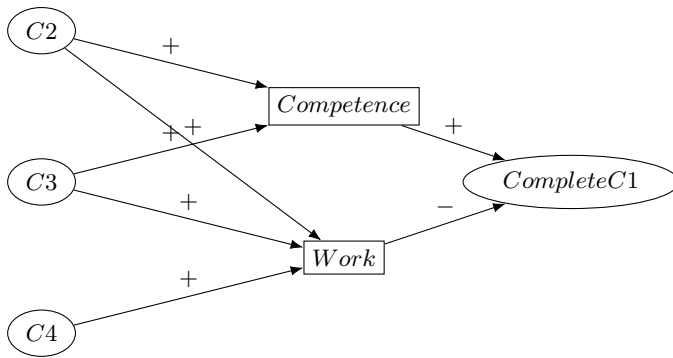


Figure 1: An example of causal modelling explaining factors influencing whether Course 1 (C1) is dropped or not.

terfactual explanations. In our case, we want to know whether not taking one or a combination of the classes will lead to a different prediction of our model. This can provide sensible advice.

3. *Understand the causal relationships among features and latent factors.* This refers to a causal understanding of features and latent factors. In this last step, we try to uncover the DAG (as shown in Figure 1) by theorizing about latent factors and testing whether the observations support this. We aim to understand that the courses influence latent factors competence and workload and which course influences which factor in which way. Not only does this lead to the best intervention for Alice; moreover, we can use this knowledge to construct better programs for all students.

If we use LIME off-the-shelf, we can extract the k most important features per person but counterfactual explanations are not provided. Thereby, we reach the first but not subsequent steps of interpretability. This limitation of XAI particularly regarding counterfactual explanations has been addressed and sometimes dealt with by other scholars as well [9, 8, 1]. Given these limitations, it is, therefore, maybe not surprising that few scholars saw it worth to employ XAI. How can we reach the other steps of interpretability and find a basis for personalized intervention? We argue that in particular reaching the third step calls for turning towards and employing techniques of social science. Firstly, we need an understanding of the concept of counterfactuals and aim to extend LIME in that way. Secondly, we need a theory-, instead of data-driven approach to explore causal mechanisms. Social science is well equipped for this task.

3.1 Reaching Step 2: MC-LIME

Reaching step 2 is easily possible as it rather naturally extends the idea of LIME, but requires an understanding of the concept of counterfactuals common in social science. In short, we attempt to answer the question of what would have happened regarding the outcome (the prediction) if the treatment (the features' values) had been different. In our case, we consider the k features extracted by LIME that have a *positive* impact on the drop-out probability (i.e. make it more likely that someone drops out). Then, we check for the smallest subset of these features that - when changing

their values - changes the prediction and return the features and changed values³. We achieve this by iteratively changing one feature's value; if this never leads to a prediction change, we check for all combinations of two features etc. Because we look for the smallest necessary change, we call this approach Minimally Counterfactual LIME (MC-LIME). If multiple subsets of the same size exist, but we only want a certain number, we can select to receive the c changes that lead to the largest difference in the output probability. This procedure is straightforward for binary variables where we can simply use the complementary value. For categorical and ordinal features, we propose iterating through all possible values; all values, for which a change is reported, are stored - the highest change counts towards the selection of the top c features. For count features, we propose shifting the value a standard deviation towards the mean, so that the change is large enough to make a substantial difference. The resulting subset tells us what would minimally need to change in order to change the prediction and how this change would need to look like.

3.2 Reaching Step 3: A Causal Analysis

While this information is already very important, it is not enough to provide good interventions and to potentially construct programs, though. In order to know why features matter - for the model but hopefully also overall - we propose to use *all* features LIME returns (positive and negative impact) as a basis for a deeper analysis. For this theory-driven approach, we propose to follow the steps:

1. Extract all features and their impacts and use it to cluster people into groups. Therefore, we only work with a subset of all extracted variables that are known to be relevant for a set of students thereby simplifying the model while at the same time assuring that we use relevant features.
2. For the demographic (and if available social and psychological) features, e.g., age, having a student job, living closer or further away from university, look for social science studies that investigate their effect on drop-out and let it inform you on causal mechanisms. If you find that other features could also be important, add them.
3. For features specific to your domain, e.g. the courses offered, try to understand what they are about and how this could influence the outcome variable, i.e. the drop-out.
4. Begin drawing a DAG and consider the following questions: (a) Is a connection between two variables direct or does it go through a latent variable we cannot observe? Does the latent variable mediate the effect? (b) Is there an actual relationship between two variables or are they confounders meaning that a third variable effects both? (c) Does a third variable moderate the effect between two variables? (d) Is the effect linear or quadratic?

³Note that this is very similar to LIME's understanding of feature importance.

5. Model a regression formula according to your DAG and run the regression on the training data with success or drop-out being the dependent (or target) variable.
6. Check the effects of the terms of your formula. What is significant? Does the direction conform to your theoretical considerations?
7. Construct personalized interventions according to social science theory and in combination with the insights of step 2.

4. EVALUATION

In order to demonstrate the pipeline and its applicability, we test our approach on an artificial and a real-life set of data.

4.1 Data

Artificial Data. As, typically, we do not know the real causal mechanisms and can only make informed guesses considering the existing literature and our own reasoning, we test MC-LIME and our causal framework on artificial data. Regarding MC-LIME we can test whether it returns the feature subset that is intended to make a difference. It is, of course, not very telling to use this data on our causal framework as we know the causal mechanisms we decided on. However, it is still valuable to see whether we can recover the intended effects and their directions as specified in data generation. Our data consists of the target variable drop-out and 26 other binary features. Of these 26 features, when they are set to 1, eight have no effect, three have no direct effect but do have one when combined with other variables, two have a negative impact on drop-out that reverses when combined with other variables, three have a positive impact that reverses when combined with other variables, five have a negative effect and five have a positive effect. The first row of Table 1 summarizes this; a plus indicates a positive effect on drop-out, a minus a negative effect. The number of symbols represents the strength of the effect, as each causal relationship was given a weight by which the probability of a drop-out changes. We created 10,000 instances by randomly sampling features. The drop-out value was determined by the sum of the weights of the non-zero variables. If the sum was 0.5 or higher, we assigned a 1, else we assigned a 0. This resulted in 30% of instances having assigned a 1.

Real-Life Data. In order to demonstrate the process on real-life data, we gathered information on a mandatory first-year theoretical computer science course - that we will call C1 - part of a three-year Bachelor degree at the University of Mannheim, Germany.⁴ We have information on all students that registered for this course between 2010-2020 and try to predict who will drop out. Note that students who failed or dropped out once and then registered in subsequent years may appear in the data twice. Our data contains 1,738 instances. Furthermore, even though the course is meant to be taken in the first year, many students take it later. The data contains seven demographic features, and 160 features on high school results, previous courses taken, previous results and drop-out behavior, and classes taken in parallel.

⁴The data was k-anonymized prior to analysis to ensure privacy.

To understand our data structure, consider a course A. This course has four features assigned to it: whether it is taken in parallel to C1, whether the student failed it, whether the student dropped out, or whether the student passed. Note that a student can have 1 assigned to several of these features, if, e.g., a person first dropped and then passed course A. Again, remember that this only encompasses the information we have at the time when the student registers for the course we are predicting on. In total, we consider 30 courses. Table 2 provides summary statistics of the data.

4.2 Step 1: LIME

For both sets of data, we predicted the drop-out using several methods: Support Vector Machine (SVM), a simple Deep Neural Network (DNN), Naive Bayes, Decision Tree, and Random Forest. Then, we selected the model leading to the highest F1-value and accuracy in the test data. For the artificial data, this was the DNN with an accuracy of 99.5% and a F1-value of 0.99. For the real data, it was the SVM with an accuracy of 87.12% and an F1-value of 0.9. Having selected the best model, we extracted the ten most important features and their directions for each instance of the test data. We only considered those instances for which drop-out was predicted, as these are the ones we are most interested in.

Artificial Data. 318 of the 1,000 test instances were predicted to drop out. Table 1 shows which features were extracted at least for one instance. We can see that feature V8 was not considered important at all, even though it is supposed to have a negative effect on drop-out. In contrast, V20 was extracted once, even though it should not have an effect. Furthermore, V17 and V18 were not extracted; these features do not have an effect on their own, but do when combined. The most extracted features were V2, V11, V12, and V13 which were extracted for each instance, followed by V5 (316), V1 (312), V6 (284), and V3 (213). Note that this does not mean that for all those predicted to drop out, each of these variables was set to 1 or had a positive impact as the table also includes features that have a negative impact on drop-out. As a matter of fact, the extracted directions of the effects are correct for all extracted features that upon being set to 1 are supposed to have a positive or negative effect on drop-out. For those features for which the direction of the effect changes upon combination with others, we can see that the reverse effect is extracted. This shows the limitations of LIME and, therefore, the importance of our remaining steps.

Real Data. For 26 instances the label drop-out was predicted. In total, 25 important features were extracted; of these, six only appeared once. The most frequently extracted features were whether a person planned to take the exam on the first date or in the resit (26)⁵, the study year (26), the age (26), whether two other first year classes were taken in parallel (24 each), whether one of these first-year courses had been dropped before (20), and whether a second year course had been passed (12).

⁵Students have the opportunity to decide between taking the exam right after the lecture period or two months later; the latter is known as the resit date.

Table 1: Artificial Data - variables, their effects, and what could be recovered using LIME and regression on extracted features.

Effects	Pos. Effect	Neg. Effect	Effect in Combination	No Effect
Variables	V1(++), V2(++), V3(++), V4(++), V5(++)	V6(--), V7(-), V8(-), V9(-), V10(-)	V11(--)+V12(--)+V13(No): +++; V14(+)+V15(+)+V16(+): ---; V17(None) + V18(None): ++	V19-V26(No)
Recovered (LIME)	V1(+), V2(+), V3(+), V4(+), V5(+)	V6(-), V7(-), V9(-), V10(-)	V11(+), V12(+), V13(+), V14(-), V15(-), V16(-)	V20(-)
Changed Prediction (MC-LIME)	V1, V2, V3, V4	V6, V9	V11, V12, V13	V20
Recovered (Regression)	V1(+), V2(+), V3(+), V4(+), V5(+)	V6 (-), V7 (-), V9(-), V10(-)	V11(-)+V12(-)+V13(None): +; V14(+)+V15(+)+V16(None): -; V17(None)+V18(None): +	V20(-)

Table 2: Summary statistics of real-life data regarding the first year course C1.

Variable	Key Statistics
Age	20.49 (min: 16, max: 36)
Year	1.7 (min: 1, max: 5)
N of Attempts	1.3 (min: 1; max: 3)
Gender	female: 18.35%, male: 81.65%
Nationality	domestic: 83.77%
Domestic HS Degree	91.48%
Drop-Out of C1	41.49%

4.3 Step 2: MC-Lime

Artificial Data. We now selected those important features that have a positive effect on drop-out for each instance and iteratively changed the values. For 302 instances, it was enough to change a single value to change the prediction; for 14 of these, there was only one feature which managed to change the prediction on its own. Table 1 displays what features changed the prediction on their own for at least one instance. 14 instances needed two changes, the remaining three changes. The feature most often leading to a change when assigned a different value was V11 (291), followed by V13 (288), 12 (287), and V2 (112). Interestingly, V20 also changed the prediction on its own once. Several variables could not change the prediction on their own. Apart from V5, though, these are only variables that have a small impact on the drop-out rate in comparison.

Real Data. Proceeding in the same fashion, we found that 14 instances only needed a change in one feature to change. This rose to 19 when we considered changes in features referring to the same course as just one. There were 10 instances for which only one specific feature changed the prediction. Two instances needed two changes, the remaining instances three or more. 16 features changed the prediction on their own for at least one instance. The instances most often leading to changes were relating to two of the three classes extracted as important before (16, 13), the variable indicating the date (16), the age (8), and semester (6). Of course, a person cannot change their age upon learning that this contributes to the prediction. However, universities can identify causal mechanisms explaining the importance of age and then construct specialized offers. In order to be able to this, we of course need to continue with step 3.

4.4 Step 3: Regression and Modelling

Artificial Data. For the artificial data, we simply used all the extracted features and our knowledge⁶ about the data generation to construct the logistic regression formula. Then, we entered this together with the training data in a logistic regression. The last row of Table 1 shows the recovered effects. All variables entered into the regression were significant (then they were given the sign of the direction of their effect in the table) apart from V16 and V11 - even V20 (albeit only on the 5%-level) which means that by chance in data generation, more instances got the label drop-out assigned which also received a 1 in this variable. V16 was positive but not significant, even though it should be. V11 was correctly identified as no longer being significant once combined with the other two variables. All effects now also had the correct directions. Interaction terms of V11, V12, and V13 and V14, V15, and V16 were also significant and had the correct direction. We can see that reasoning about and investigating causal mechanisms made it possible to recover most effects and their directions.

Real-Life Data. For the real-life data, as explained above, we first clustered our test instances using k-Nearest Neighbor based on the features extracted so that we only focus on features relevant for this set of students. We chose $k = 3$ upon visual inspection. For illustrative and space reasons, we will focus on the largest of the clusters containing twelve instances. We used all features that were extracted for more than one instance in the cluster (Table 3). The only two features that are not specific to our setting are the variables age and nationality. Therefore, we consult the literature on these variables. For age, scholars are divided. While some studies stress that older students are generally more successful and achieve higher grades, others find that advances in age can also be seen as a positive predictor for drop-out [16, 18, 4]. The former is generally attributed to older students being more certain of their goals and having an increased focus; the latter is often attributed to having other important parts of life such as a family or a job. Based on this, we theorize that those being a little older are influenced by the former, and those who are much older by the latter; thus, for the regression, we include *age* and *age*². Of course, that age has been selected in the first place, could also be due to the fact that those students taking the class later in their studies are older and may regularly struggle with courses. To account for this, and because it is also extracted as a feature, the number of years one has studied is also included. For domestic nationality, the likely reason

⁶Therefore also entering V17 and V18 again.

for the negative effect LIME implies is that the degree is in German which creates a language barrier to non-German speaking students [12, 7]. This might be mitigated when a student already received their high school diploma in Germany; thus, we enter this variable in an interaction term, even though, it is not extracted as an important feature. The other features are specific to our setting. We argue that having failed the course before leads to a decreased probability of dropping out because students have already completed the course before. Writing the exam on the resit date leads to an increased probability of dropping out because the exam is written almost two months after the end of lectures meaning that a) students may have not paid enough attention on this course during the lecture period and b) students may have forgotten important information in the meantime. We argue that having passed courses C2, C3, and C4 leads to a decreased drop-out probability, because these courses have connected contents and require a similar skill-set. Those who did not struggle much with these courses can then also complete this one. Likewise, having struggled in these courses leads to a higher drop-out probability. Furthermore, we argue that taking C2 and C3 together with our queried course C1 - as intended by the study program - may lead to a very high workload; thus, we also include an interaction term of these. Table 3 shows our results, the middle column summarizes the results of the logistic regression without interaction or quadratic terms, the right-hand column the results including these terms. The symbol “+” indicates a positive effect on the drop-out probability, symbol “-” a negative one. One “+” indicates an effect on the 10%, two on the 5%, and three on the 1% level; a “No” indicates no significant effect. We can see that the effect for age - which is at first positive - reverses when age^2 is added with higher age now leading to a decreased drop-out probability on average, though age^2 is not itself significant most likely due to the small sample size. Similarly, the non-significant effect for domestic students and high school diploma may be due to that. The study year does not matter, but the date greatly matters with taking the exam at a later date leading to a higher probability on average. Having failed the course before, leads to a smaller drop-out probability, but having dropped it to a larger one. Only taking C3 in parallel seems to have an effect on its own. C2 has no effect. For C3 we see that those who dropped this also have a higher probability of dropping C1; having passed C3 also means that C1 will likely be completed. For C4, even not having passed already leads to a smaller probability of drop-out. Having passed all three courses also leads to a smaller drop-out probability. What do we take away from this? Generally, we should investigate whether the age effect that comparatively old and young students struggle persists across the overall program. If so, we should think about how to help these age groups. Furthermore, considering this course in particular, we should encourage the students to not choose the resit date. We should also identify students who have struggled with the courses of similar content before and offer increased assistance and attention to them. How to best do this, is something social science may also help us with. Finally, we combine our insights of step 2 and 3 for each student to tailor the best intervention.

5. CONCLUSIONS, LIMITATIONS, & FUTURE WORK

Table 3: Regression results of real-life data.

Variable	Without Interaction	With Interaction
Age, Age^2	+	-, No
$Year, Year * Age$	No	No, No
$Date$	+++	+++
$Domestic, Domestic * HS$	No	No, No
$failC1, dropC1$	No, +	-, +
$parC2, parC3, parC4, parC2 * parC3$	No (all)	No, -, No, No
$failC2, dropC2, pasC2$	No(all)	No(all)
$failC3, dropC3, passC3$	No, +, No	No, +, -
$failC4, dropC4, passC4$	-, No, --	-, No, --
$passC2 * passC3 * passC4$		-

In this paper, we addressed the issue of a lack of XAI in EDM. More specifically, we attempted to provide a framework enabling us to use the information extractable from predictions as a basis for a personalized intervention system. To highlight the challenges and requirements, we came up with a step-wise model of interpretability where step 1 means identifying important features, step 2 identifying the minimal set of value changes to change the prediction, and step 3 identifying causal mechanisms. We described methods to reach each of the steps and evaluated them on an artificial and real-life dataset showing the applicability. Our results on artificial data showed that the method works well when we correctly theorize about causal mechanisms. Of course, we may not always be able to do that. This is a limitation of our work which, in general, provides no “one size fits all”-formula, but needs to be adjusted for different settings. Furthermore, our methods can certainly be further refined but we hope that our step-wise model of interpretability provides a good orientation. A third limitation is that we do not show an actual application of our method in a real-life setting meaning that we cannot evaluate whether the conclusions derived from our analysis benefit the students in practice. This is a future endeavour. When using predictive systems in practice, we would like to once again stress that this should be made clear to students and should be very transparent. Finally, we would like to argue for an increased collaboration among social and computer scientists. Whereas computer scientists are typically experienced with predictions and deriving knowledge from data, they lack experience when it comes to theory-driven approaches and causal analysis. Social scientists, in contrast, typically do not work on predictions but are knowledgeable regarding statistical tools to uncover causal mechanisms and deriving models from theory. While these differences in approaches are prone to hinder collaboration, for this task it will greatly benefit both disciplines. Furthermore, while social science informs our models, it can also gain new insights through large-scale predictions and deriving information from data.

6. REFERENCES

- [1] A. Adadi and M. Berrada. Peeking inside the black-box: a survey on explainable artificial intelligence (xai). *IEEE access*, 6:52138–52160, 2018.
- [2] R. Alamri and B. Alharbi. Explainable student performance prediction models: a systematic review. *IEEE Access*, 2021.

- [3] M. Baranyi, M. Nagy, and R. Molontay. Interpretable deep learning for university dropout prediction. In *Proceedings of the 21st Annual Conference on Information Technology Education*, pages 13–19, 2020.
- [4] R. Chen. Institutional characteristics and college student dropout risks: A multilevel event history analysis. *Research in Higher Education*, 53(5):487–505, 2012.
- [5] M. Chitti, P. Chitti, and M. Jayabalan. Need for interpretable student performance prediction. In *2020 13th International Conference on Developments in eSystems Engineering (DeSE)*, pages 269–272. IEEE, 2020.
- [6] F. Del Bonifro, M. Gabbrielli, G. Lisanti, and S. P. Zingaro. Student dropout prediction. In *International Conference on Artificial Intelligence in Education*, pages 129–140. Springer, 2020.
- [7] S. Evans and B. Morrison. Meeting the challenges of english-medium higher education: The first-year experience in hong kong. *English for Specific Purposes*, 30(3):198–208, 2011.
- [8] M. T. Keane, E. M. Kenny, E. Delaney, and B. Smyth. If only we had better counterfactual explanations: Five key deficits to rectify in the evaluation of counterfactual xai techniques. *arXiv preprint arXiv:2103.01035*, 2021.
- [9] M. T. Keane and B. Smyth. Good counterfactuals and where to find them: A case-based technique for generating counterfactuals for explainable ai (xai). In *International Conference on Case-Based Reasoning*, pages 163–178. Springer, 2020.
- [10] S. M. Lundberg and S.-I. Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.
- [11] R. Manrique, B. P. Nunes, O. Marino, M. A. Casanova, and T. Nurmikko-Fuller. An analysis of student representation, representative features and classification algorithms to predict degree dropout. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge*, pages 401–410, 2019.
- [12] L. Morrice. Refugees in higher education: Boundaries of belonging and recognition, stigma and exclusion. *International Journal of Lifelong Education*, 32(5):652–668, 2013.
- [13] B. Prenkaj, P. Velardi, G. Stilo, D. Distanto, and S. Faralli. A survey of machine learning approaches for student dropout prediction in online courses. *ACM Computing Surveys (CSUR)*, 53(3):1–34, 2020.
- [14] L. Qiu, Y. Liu, Q. Hu, and Y. Liu. Student dropout prediction in massive open online courses by convolutional neural networks. *Soft Computing*, 23(20):10287–10301, 2019.
- [15] M. T. Ribeiro, S. Singh, and C. Guestrin. Model-agnostic interpretability of machine learning. *arXiv preprint arXiv:1606.05386*, 2016.
- [16] T. M. Spitzer. Predictors of college success: A comparison of traditional and nontraditional age students. *Journal of Student Affairs Research and Practice*, 38(1):99–115, 2000.
- [17] W. Xing and D. Du. Dropout prediction in moocs: Using deep learning for personalized intervention. *Journal of Educational Computing Research*, 57(3):547–570, 2019.
- [18] D. Yasmin. Application of the classification tree model in predicting learner dropout behaviour in open and distance learning. *Distance Education*, 34(2):218–231, 2013.
- [19] H. Zeineddine, U. Braendle, and A. Farah. Enhancing prediction of student success: Automated machine learning approach. *Computers & Electrical Engineering*, 89:106903, 2021.

An Evaluation of code2vec Embeddings for Scratch

Benedikt Fein, Isabella Graßl, Florian Beck, Gordon Fraser
University of Passau
Passau, Germany

ABSTRACT

The recent trend of embedding source code for machine learning applications also enables new opportunities in learning analytics in programming education, but which code embedding approach is most suitable for learning analytics remains an open question. A common approach to embedding source code lies in extracting syntactic information from a program's syntax tree and learning to merge these into continuous distributed vectors (e.g., CODE2VEC). CODE2VEC has been predominantly investigated in the context of professional programming languages, but learning analytics are particularly important in the context of educational programming languages such as SCRATCH. In this paper, we therefore instantiate the popular embedding approach CODE2VEC for SCRATCH programs, create three different classification tasks with corresponding datasets, and empirically evaluate CODE2VEC on them. Our experiments demonstrate that a transfer of CODE2VEC to the educational environment of SCRATCH is feasible. Our findings serve as a basis to apply code embeddings to further educational tasks such as automated detection of misconceptions of programming concepts in SCRATCH programs.

Keywords

code2vec, Scratch, programming education.

1. INTRODUCTION

The application of natural language processing (NLP) and machine learning (ML) methods in the field of software engineering (SE) is gaining popularity in research and industry [27]. A central prerequisite for such machine learning applications on source code is to represent semantically similar code as similar continuously distributed vectors, the *code embeddings*, in a vector space. Popular code embeddings such as CODE2VEC [6] have been successfully used for program analysis tasks such as predicting method and variable names, or identifying bugs and misconceptions [3, 5, 17].

B. Fein, I. Graßl, F. Beck, and G. Fraser. An evaluation of code2vec embeddings for Scratch. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 368–375, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853103>

Programming education research frequently relies on analysis of learners' programs, for example to automatically detect incorrectly used programming concepts and bugs [2, 11, 25, 26]. Code embeddings bring the promise of novel applications also in the educational domain [9, 10, 24]; e.g., continuously distributed vectors make it possible to monitor learner trajectories or to detect outliers and anomalous behavior. However, code embeddings are predominantly generated from syntactic features of the source code. For example, CODE2VEC considers the relation of pairs of textual tokens in the context of the syntax tree that results from parsing the source code. Most code embedding approaches are designed for textual programming languages such as Java or Python. Programming education, however, is frequently based on simplified block-based programming languages such as SCRATCH [22]. These programming languages are intentionally designed to reduce the syntactic overhead for learners, and may thus affect the same syntactic properties of programming languages that make them amenable to code embedding models. This may in turn affect the applicability of these models in an education context.

The aim of this paper is to adapt and investigate the popular CODE2VEC code embeddings for the educational programming language SCRATCH. We implement an analysis for SCRATCH programs that extracts the path context information on which CODE2VEC is built. We then create three different classification tasks with corresponding datasets to study the suitability of the resulting embeddings:

- Girls and boys are known to implement different project types and programming concepts [13, 16]; we explore whether code embeddings can capture these nuances.
- A major characteristic of SCRATCH programs with educational implications [1] is their type (e.g., game, animation, etc.). We explore whether code embeddings enable the prediction of project types from code.
- The original evaluation of CODE2VEC explored the ability of embeddings to capture semantic content by predicting names of methods. We adapt this task to SCRATCH by predicting names of sprites.

Although SCRATCH code differs from text-based code in important ways affecting code embeddings, such as the structure or size of syntax trees, or the organisation into sprites and scripts rather than classes and methods, we find that CODE2VEC nevertheless performs well at these tasks.

2. BACKGROUND AND RELATED WORK

To understand the application of CODE2VEC to the introductory programming language SCRATCH, this section outlines the concepts and their use cases.

2.1 The Scratch Programming Language

SCRATCH is a block-based programming environment that is particularly designed for learners due to its ease of use through the arrangement of visual blocks [22]. In SCRATCH, the behavior of graphical objects, the sprites, is controlled by means of code blocks, which are assembled to scripts. The code blocks have particular shapes so that they can only be assembled in syntactically valid ways, without the need for the syntactic overhead of text-based programming languages (such as indentation, braces, semicolons, etc.) Code blocks control the appearance and behavior of sprites, as well as interactions with the user.

Besides the intuitive programming user interface, the popularity of SCRATCH is also supported by a rich ecosystem of users sharing their programs publicly and interacting around them. In addition to accessing this information through the user interface, it is also possible to use a REST-API to programmatically access all publicly available data conveniently, which is helpful to enable data mining applications.

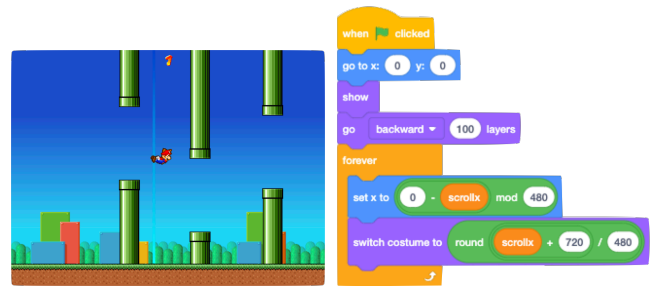
SCRATCH programs are categorized into one or more project types: *games*, *stories*, *animations*, *music*, *art*, and *tutorials*. It has been established that some project types require certain programming concepts more than others [1,18]. Furthermore, it has been repeatedly observed [13,16] that there are gender-dependent preferences regarding the project type and thus in the programming concepts: While girls mainly prefer programs with storytelling elements, boys implement more programs with game structures [1,13,16].

2.2 Analyzing Scratch Programs

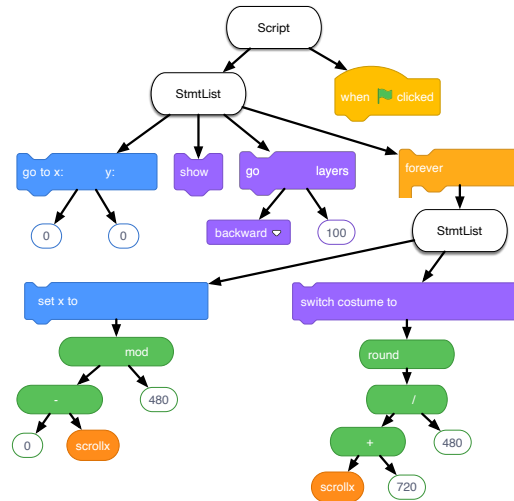
The source code of programs in text-based programming languages is represented using plain text files. In contrast, block-based programs require an intermediate format to describe the program blocks. In particular, SCRATCH programs are represented using JavaScript Object Notation (JSON) format. These JSON files organize programs in terms of their “targets” (stage and sprites), and for each target the JSON file lists its name, its procedures (i.e., custom blocks), scripts, variables, lists, messages, sounds, costumes, and blocks. The blocks are organized as lists, where each element contains a unique identifier as well as the identifiers of the parent and successor blocks, we well as any parameter blocks. Whereas text-based programs are often used directly as input for machine learning approaches, this JSON format is intuitively less suitable for NLP-based approaches.

Static program analysis is usually not conducted on the raw text representation, but the abstract syntax tree (ASTs) intermediate representation, which results from parsing the source code. An AST-like representation is used by the SCRATCH virtual machine in order to interpret SCRATCH programs. The LITTERBOX [12] analysis framework provides a Java API to parse SCRATCH programs and apply static analysis. Figure 1 shows a publicly shared example project¹

¹<https://scratch.mit.edu/projects/18024798>



(a) The game in action. (b) Code of the level sprite.



(c) Abstract syntax tree of the script in Fig. 1b.

Figure 1: Example project (ID: 18024798): Flappy mario.

implementing a flappy bird game (Fig. 1a). Figure 1b shows the code of one of its 24 sprites: This sprite represents the ground and the script implements the scrolling motion to simulate movement of the “flappy Mario” character. Figure 1c shows the AST representing the same script: Although this AST is slightly simplified for space reasons, it is noteworthy that this AST is less “abstract” than an AST for other languages would be. For example, while a text-based programming language would likely define an abstract token type for binary operators, with the actual operator as one of its leaf children², in SCRATCH none of the operators are leaves, while only variables, literals, menu-options (e.g., `backward ▾`), and blocks without parameters (e.g., `show`) appear as leaves in the AST. The AST can be used for analysis tasks such as identifying bugs [11], code smells [15], evidence of misconceptions [2], or progress and understanding [19].

2.3 Code2vec Code Embeddings

CODE2VEC [6] learns code embeddings from the syntactical representation of programs through a neural network, where semantically similar code snippets, which are implemented differently but serve the same purpose, represent vectors with a small distance to each other in the vector space. As a basis, CODE2VEC extracts *path contexts* from the AST: A path context consists of two leaves together with the path

²For example, <https://javaparser.org/>

that connects them. For example, consider the two `scrollx` variable tokens in Fig. 1c, which are connected by a path that ascends from the leaf node up to the abstract `StmntList` node, which is the least common ancestor of the two leaves, and then descends to the other leaf:



CODE2VEC extracts the path contexts for all pairs of leaves in the AST [17]. Then, a neural attention model is used to combine the path contexts to a single vector representation, i.e., the code embedding [6]. The attention mechanism learns to assign weights to path contexts depending on their importance to the semantics of the code snippet, which is assumed to be captured by method names. Consequently, CODE2VEC is applied to individual functions; note that, except for custom blocks, SCRATCH scripts are not named. The final single vector that represents the code is calculated as a weighted sum over the learned individual vectors for the path contexts [6]. When given an unseen code snippet, the network can then use the learned weights of the paths to calculate such a weighted sum again and therefore assigns a similar vector to semantically similar program code.

2.4 Code Embeddings in CS Education

Various approaches to create code embeddings have recently been considered in an education context. Piech et al. [21] created embeddings for programs written in a text-based educational language by executing unit tests; these embeddings were shown to be useful for predicting which students would benefit from instructor feedback. Azcona et al. [7] demonstrated that CODE2VEC embeddings on Python code are particularly promising on learner’s code when compared to word embeddings applied directly to tokens. Cleuziou et al. [9] proposed a two-step embedding approach where first the AST paths executed by predefined test cases are extracted, and embeddings are created using document embedding techniques. This approach was applied to Python code for the task of propagating teacher feedback. Shi et al. [23] evaluated the two code embedding techniques CODE2VEC and ASTNN [28] for the supervised learning task of bug prediction on Java programs. Paassen et al. [20] introduced the AST2VEC approach for embedding Python programs, with the aim to also support transformations back from embeddings to source code. Finally, Bazzocchi et al. [8] proposed to bypass the embedding problem by using an encoder-decoder architecture directly on Python source code. All of these approaches have in common that they are applied to text-based programming languages.

To the best of our knowledge, there is only one prior investigation of CODE2VEC on block-based programs: Shi et al. [24] applied CODE2VEC to SNAP [14] and clustered the embedded programs to identify clusters representing common misconceptions. Shi et al. demonstrated that for this application CODE2VEC embeddings are superior to other models of the code, such as Bags of Words. In this paper we aim to shed more light on how CODE2VEC generalizes to other tasks. Although SNAP represents programs using XML files that are closer in their structure to regular programs, the resulting ASTs are similar to those of SCRATCH, and so we expect our findings to generalize also to SNAP.

3. METHOD

To evaluate the CODE2VEC code embeddings for SCRATCH programs, we investigate the following research questions:

- RQ 1** Gender: How accurately can CODE2VEC assess a binary classification task on SCRATCH programs?
- RQ 2** Category: How accurately can CODE2VEC assess a multi-class classification task on SCRATCH programs?
- RQ 3** Sprite naming: How accurately can CODE2VEC assess a classification task on SCRATCH programs?

3.1 Datasets

The RQs require different datasets for their classification tasks: predicting gender, project type, and sprite names.

RQ1. To answer RQ1, we use a dataset of 317 SCRATCH programs [13], of which 171 were created by 64 (self-identified) girls and 146 by 68 boys in the range of 8–10 years. The programs are the result of the final task of a multi-day introductory programming course; the children were tasked to implement a SCRATCH program based on a topic of their own choice. The resulting programs were then manually labelled with the students’ genders. The programs of both genders are comparable in block size (on average: boys 27, girls 22) and number of sprites (on average: girls 6.10, boys 4.78) although the types of blocks and sprites differ [13].

RQ2. To answer RQ2, we sampled 216 000 SCRATCH programs publicly shared between March 2021 and June 2021. Since the REST API of the SCRATCH website³ does not provide information about project types, we downloaded programs from each category individually by using GET requests containing certain category names⁴. To create a balanced dataset we subsampled these programs to create a uniform distribution of labels; each program can belong to one or more categories. Since users often use hashtags with all category keywords to gain more visibility, the dataset contains a high percentage of misclassifications. To mitigate these misclassifications, we applied several filtering steps: First, we excluded duplicates and remixed programs. We then also excluded programs tagged as games from the music and tutorial categories, as users often incorrectly add the hashtag music to their game programs simply because they contain background music. In addition, we removed programs in the tutorial, art, music categories that contain their category keyword in the notes and credits section, as users would state credits to the music they included. We evaluated the effectiveness of these filtering steps by manually classifying 10 randomly selected programs from each category, which confirms a decrease of the misclassification rate to 20 % or less in every category. The final dataset consists of 50 560 multi-labelled SCRATCH programs in 40 categories representing various combinations of the six base-categories.

RQ3. To answer RQ3, we created a randomized sample of 530 696 SCRATCH programs publicly shared between April 2007 and April 2020. The data mining was realized by retrieving the 10 000 most recently publicly shared SCRATCH programs each day using the REST API of the SCRATCH website in the mentioned period.

³<https://github.com/LLK/scratch-rest-api/wiki>

⁴<https://scratch.mit.edu/explore/programs/all/>

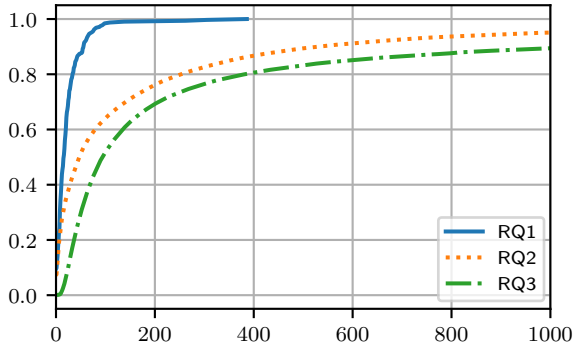


Figure 2: Cumulative distribution of block counts.

3.2 Data Analysis

Each dataset is divided into training, validation and test dataset with a ratio of 80:10:10. For RQ1, the training set contains 253 programs, the test and validation set 32 programs each; for RQ2 the training set contains 34639 programs, the test and validation set 4335 programs each. To answer RQ3, we use a classification task to identify the names of sprites based on their code, thus resembling the method name prediction task [6]. In contrast to RQ1/RQ2, this task considers the ASTs of individual sprites, rather than entire programs. The training set contains 504503 programs with 4487940 sprites, the test set 15000 programs with 137429 sprites and the validation set 15000 programs with 132875 sprites. The training dataset contains 247317 different names with 90802 of them appearing more than once. The 100 most frequent names are used for 580544 sprites. We use accuracy, precision, recall and F1-score to quantify the performance of the generated models. To better understand the contribution of the code structure versus the literals used in programs, we conduct a small ablation study with a model for each task where literal values are replaced with abstract tokens for their type (string or number).

3.3 Data Preprocessing

The SCRATCH programs must first be processed to extract the path contexts in an appropriate format for the CODE2VEC model. SCRATCH programs are saved as .sb3 files, containing image and audio files as well as the JSON program code. We use LITTERBOX [12] to parse these JSON files into their AST representation. We extended LITTERBOX with the extraction and cleanup of the path contexts, such that no additional intermediate representations of the graph structure are needed. The extraction of path contexts ignores non-code related aspects of the AST, such as the positions of blocks in the code editor or post-it style comments.

For RQ1 and RQ2, the entire AST of the program, starting with the Program root node, is considered when extracting path contexts, and the labels are included in the dataset. For RQ3, we extract the path contexts per sprite from their sub-trees (ActorDefinition nodes in LITTERBOX), as well as the sprite name as the label for the classification task. Similar to how CODE2VEC treats method names, sprite names are split on special characters into subtokens, and the subtokens are normalized to only contain lowercase letters. The final sprite name is then obtained by joining the non-empty

Table 1: Hyperparameters used for Java code [6] compared to the ones for Scratch experiments.

	Java	RQ1	RQ2	RQ3
number of contexts	200	200	1000	200
embedding size	128	128	128	128
max path length	8	8	12	8
dropout keep rate	0.75	0.75	0.75	0.75
batch size	1024	16	512	1024

subtokens back together with a vertical bar “|” as separating character to support manual interpretation. Additionally, there can be sprites that have the default name (depending on the language settings, e.g., “sprite”) after this normalization step. These are sprites that were not named by the user, and therefore the name cannot be assumed to describe the code. We excluded these sprites from the dataset.

3.4 Neural Network Structure

For all experiments we used the network structure as described by Alon et al. [6] and their implementation⁵. Even after extensive hyperparameter tuning by rerunning the experiment while iteratively changing the parameters one at a time, most of the values as used by Alon et al. for their analysis on Java code [6] also perform best on SCRATCH code (see Table 1). Consequently, we mainly re-used the default or similar values for common hyperparameters. We adapted batch sizes for the different experiments based on the dataset sizes: For the small dataset for RQ1 we reduced the batch size to 16; for RQ2 we used a batch size of 512.

Of the additional hyperparameters specific to the domain of code embeddings, the maximum considered path length and the number of path contexts used for the representation require particular consideration: Increasing the maximum path length allows the model to learn about related elements that are further apart in the source code. However, this also increases the number of generated path contexts. Due to the limited amount of memory available to us during the training phase, a random sample of those has to be chosen. By generating too many path contexts, the chance of missing semantically important ones during sampling increases.

Generally, the maximum path length that should be considered in the case of SCRATCH is higher than for the original Java method name experiment: Even a single sprite encapsulates the full behavior of a figure in a game and can contain multiple scripts, each controlling different aspects of behavior. Therefore, a sprite can be seen as comparable to a class in Java with scripts corresponding to methods. This results in long paths especially for connections between AST leaves placed in different scripts or sprites. Figure 2 shows the average program sizes for the three different datasets, showing that the RQ2 and RQ3 datasets have substantially larger programs than the gender classification task (RQ1). As the project categorization task (RQ2) considers entire programs, only 2% of all paths would be retained when pruning at the maximum of eight, as used in the original Java study (Fig. 3). Consequently, for RQ2 we increased the length to 12, resulting in 18200 path contexts.

⁵<https://github.com/tech-srl/code2vec>

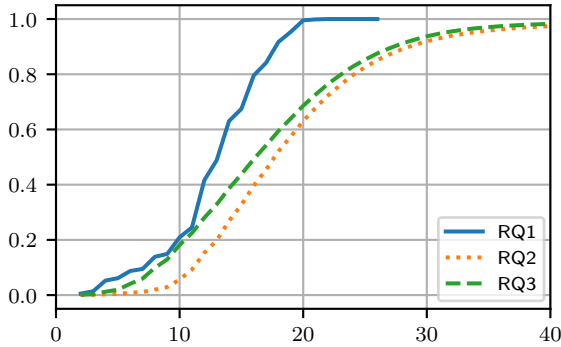


Figure 3: Cumulative distribution of path lengths.

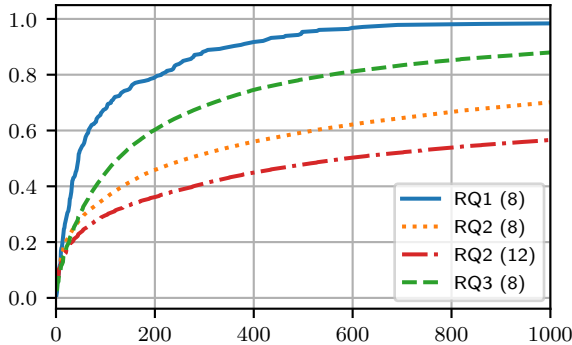


Figure 4: Cumulative distribution of path context counts. Maximum path length hyperparameter in parentheses.

Figure 4 shows the trade-off between increasing the maximum path length and the number of programs for which sampling the path contexts is necessary. Limited by graphics card memory, the model was allowed to use up to 1000 path contexts which allows us to include all of them for nearly 60% of all programs. Lowering the number of considered path contexts showed worse results during hyperparameter tuning. For RQ1, the default value of 200 was sufficient due to the small size of the programs. For RQ3, the default value of a maximum path length of eight combined with a maximum count of 200 also yielded the best results; the average number of path contexts in this dataset (1319) is significantly smaller compared to the one for RQ2.

3.5 Threats to Validity

Although our experiments aim to improve external validity by investigating CODE2VEC on three different SCRATCH tasks, results may not generalize to other tasks and embeddings (e.g., [4, 28]). Although we applied methods to ensure data quality, additional filtering may further improve results. To decrease the influence of the random initialization of internal model parameters on the small RQ1 dataset, we re-ran the experiment for each hyperparameter setting multiple times with/without reshuffling of the training set. We performed incremental hyperparameter tuning using a validation set not used during training and are reporting the results on a separate test set. Nevertheless, on different datasets other hyperparameters might yield better results. To support independent validation, our code is open source⁶, and data is available on request.

⁶<https://github.com/se2p/litterbox>

Table 2: Top-1 and top-5 accuracy, precision, recall, and F1-Score for code2vec when replacing literal values with abstract tokens (AT) and when keeping them.

Task\Metric	Prec.	Recall	F1	Acc.	Top-5 Acc.
RQ1 (AT)	78.1	78.1	78.1	78.1	—
RQ1	90.6	90.6	90.6	90.6	—
RQ2 (AT)	63.3	59.2	61.2	57.9	93.4
RQ2	64.1	60.0	62.0	58.9	93.6
RQ3 (AT)	45.4	41.6	43.4	41.5	51.9
RQ3	57.4	53.5	55.3	53.8	61.2

4. RESULTS

To evaluate the CODE2VEC embeddings for SCRATCH, Table 2 shows the performance of the CODE2VEC model on three different classification tasks.

4.1 RQ1: Gender Classification

The gender classification task shows a very high accuracy of 90.6%, suggesting that the projects are quite homogenous within the two gender groups. Grassl et al. [13] observed structural differences between the projects of the two genders, which is reflected by the high accuracy. For example, boys tend to produce interactive projects using event handling blocks and loop control structures, while girls produce more sequential programs. We observe a sharp drop in accuracy when ignoring literals (Table 2); we conjecture that this is also related to the reported sequential nature of the girls’ projects: Girls tend to produce story-like projects where sprites speak more, thus using more string literals.

RQ1 Summary. CODE2VEC is able to predict the gender based on code with a high accuracy of 90.6%.

4.2 RQ2: Project Type Classification

Compared to the gender classification task, the project category classification task shows a substantially lower accuracy of 58.9% (Table 2). The lower accuracy is likely influenced by the more challenging multi-class classification task, more noise in the data compared to the small gender dataset, and the generally larger projects used in this dataset.

While the accuracy is lower, it is comparable to the performance of the original analysis by Alon et al. [6], which was applied to individual methods. The results on the project category task thus confirm that CODE2VEC can also be applied to whole SCRATCH programs. We initially assumed that the model requires more path contexts to be able to extract information from the larger scope of the whole program. However, changing the maximum number of contexts to values between 100 and 1000 did not impact the prediction quality. In all cases the accuracy remained between 56.7% and 58.9%. We assume that the model does not actually use the path contexts as such to categorize the programs but instead focuses on the presence or absence of certain block types within the path contexts. For example, the dataset contains the categories “animations”, “games”, and “music”. Games obviously contain many blocks based around the user’s interaction with the program, whereas animations rarely do. Similarly, musical programs can be iden-

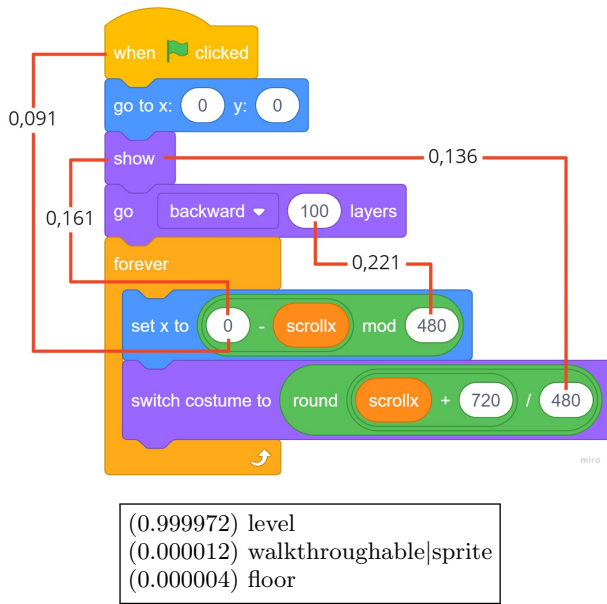


Figure 5: Example prediction with the top-4 paths with the highest attention weights, and the top-3 predictions.

tified by containing many sound-related blocks. As long as even the small path context samples contain some of those blocks distinctive to a program type, the model appears to have enough information to predict the correct category.

This conjecture is supported by the results without literals (Table 2), which even slightly increases the accuracy. This could be caused by two possible factors: The literal values might not be distinctive for project types; e.g., the movement of sprites in both animations and games relies on similar bounds checks on the visible stage area. Alternatively, some literal values are at least somewhat distinctive for the project type, but the attention mechanism focuses on other more significant differences. In both cases the model uses the attention mechanism to increase the weight for paths that contain project type specific blocks instead of relying on their start and end values. This coincides with our other hypothesis about the required number of path contexts.

RQ2 Summary. The model is able to extract semantic information from whole programs and is able to predict the project type with nearly 60% accuracy.

4.3 RQ3: Sprite Name Classification

The sprite naming task is most similar to the task used by Alon et al. [6] in the initial CODE2VEC evaluation, i.e., method name prediction. Alon et al. report F1 scores of slightly below 60%, and our results are quite close with an F1 score of 55.3% (Table 2). We conjecture that the slightly lower score is a result of the sprite naming task being slightly more challenging, as a sprite can consist of multiple scripts, and the name likely carries less semantic information than a descriptive method name.

To demonstrate that the embeddings actually represent semantic information, Table 3 shows example words and the

Table 3: Closest terms in the vector space for the example words “game”, “mario”, “easy” and “sound”.

game	mario	easy	sound
profile	luigi	hard	music
controls	link	medium	music player
text	sonic	insane	sounds
word	wario	extreme	audio
jimmy	yoshi	impossible	sfx

five closest words in the embedding space. All the terms close to “game” clearly have a connection to games themselves: Games tend to have player “profiles”, players interact using “controls”. Similarly, the terms close to “mario” mostly represent other characters from the Super Mario universe.

Unlike the project category task, the literals do contribute to some degree to the performance of the classification (Table 2). For example, Fig. 5 visualizes the most important paths in the “level” sprite from Fig. 1 as determined by the attention mechanism. In particular, the neural network gives the path between the tokens “100” and “480” the most attention; the number 480 represents the width of the stage, and thus is likely to be used in similar contexts.

RQ3 Summary. CODE2VEC can predict sprite names with a top-5 accuracy of more than 60%, suggesting that semantic information is successfully captured.

5. CONCLUSIONS AND FUTURE WORK

Code embeddings are a trending approach for program analysis, and the computer science education community has recently joined this trend and is exploring novel applications in learning analytics. An important prerequisite for applying machine learning methods is a better understanding of the capabilities and limitations of such approaches.

In order to contribute to such an improved understanding, we evaluated the popular code embedding method CODE2VEC. This is the first application of CODE2VEC to the SCRATCH programming language, and our work has identified a number of important differences between regular, text-based programming languages, and block-based languages like SCRATCH, such as differences in named entities (e.g., classes or methods) and the overall structure of the resulting AST.

Our experiments on three different classification tasks, predicting gender, project type, and sprite names, suggests that the adaption of CODE2VEC to the educational domain of SCRATCH is highly feasible, but there is room for improvement. This suggests that future work should investigate alternative code embedding methods, both those based on syntax (e.g., [28]) or graph neural networks [4].

Acknowledgements

This work is supported by the Bayerische Forschungsförderung (AZ-1520-21, “DeepCode”) and the Federal Ministry of Education and Research (01JA2021, “primary::programming”), as part of the “Qualitätsoffensive Lehrerbildung”, a joint initiative of the Federal Government and the Länder. The authors are responsible for the content of this publication.

6. REFERENCES

- [1] J. C. Adams and A. R. Webster. What do students learn about programming from game, music video, and storytelling projects? In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education - SIGCSE '12*, page 643, Raleigh, North Carolina, USA, 2012. ACM Press.
- [2] E. Aivaloglou and F. Hermans. How Kids Code and How We Know: An Exploratory Study on the Scratch Repository. In *Proceedings of the 2016 ACM Conference on International Computing Education Research, ICER '16*, pages 53–61, New York, NY, USA, 2016. ACM.
- [3] M. Allamanis, E. T. Barr, C. Bird, and C. Sutton. Suggesting accurate method and class names. In *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering*, pages 38–49, Bergamo Italy, Aug. 2015. ACM.
- [4] M. Allamanis, M. Brockschmidt, and M. Khademi. Learning to represent programs with graphs. Technical Report MSR-TR-2017-44, November 2017.
- [5] U. Alon, M. Zilberstein, O. Levy, and E. Yahav. A general path-based representation for predicting program properties. In *Proceedings of the 39th ACM SIGPLAN Conference on Programming Language Design and Implementation*, pages 404–419, Philadelphia PA USA, June 2018. ACM.
- [6] U. Alon, M. Zilberstein, O. Levy, and E. Yahav. Code2vec: Learning distributed representations of code. *Proc. ACM Program. Lang.*, 3(POPL):1–29, Jan. 2019.
- [7] D. Azcona, P. Arora, I.-H. Hsiao, and A. Smeaton. User2code2vec: Embeddings for profiling students based on distributional representations of source code. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge*, pages 86–95, 2019.
- [8] R. Bazzocchi, M. Flemming, and L. Zhang. Analyzing cs1 student code using code embeddings. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, pages 1293–1293, 2020.
- [9] G. Cleuziou and F. Flouvat. Learning student program embeddings using abstract execution traces. page 11, 2021.
- [10] A. Emerson, A. Smith, F. J. Rodriguez, E. N. Wiebe, B. W. Mott, K. E. Boyer, and J. C. Lester. Cluster-Based Analysis of Novice Coding Misconceptions in Block-Based Programming. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, pages 825–831, Portland OR USA, Feb. 2020. ACM.
- [11] C. Frädriich, F. Obermüller, N. Körber, U. Heuer, and G. Fraser. Common Bugs in Scratch Programs. In *Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education*, pages 89–95, Trondheim Norway, June 2020. ACM.
- [12] G. Fraser, U. Heuer, N. Körber, F. Obermüller, and E. Wasmeier. Litterbox: A linter for scratch programs. In *2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)*, pages 183–188, 2021.
- [13] I. Graßl, K. Geldreich, and G. Fraser. Data-driven Analysis of Gender Differences and Similarities in Scratch Programs. In *The 16th Workshop in Primary and Secondary Computing Education*, pages 1–10, 2021.
- [14] B. Harvey, D. D. Garcia, T. Barnes, N. Titterton, D. Armendariz, L. Segars, E. Lemon, S. Morris, and J. Paley. Snap!(build your own blocks). In *Proceeding of the 44th ACM technical symposium on Computer science education*, pages 759–759, 2013.
- [15] F. Hermans and E. Aivaloglou. Do code smells hamper novice programming? A controlled experiment on Scratch programs. In *2016 IEEE 24th International Conference on Program Comprehension (ICPC)*, pages 1–10. IEEE, 2016.
- [16] H.-m. J. Hsu. Gender Differences in Scratch Game Design. In *2014 International Conference on Information, Business and Education Technology (ICIBET 2014)*. Atlantis Press, Feb. 2014.
- [17] V. Kovalenko, E. Bogomolov, T. Bryksin, and A. Bacchelli. PathMiner: A Library for Mining of Path-Based Representations of Code. In *2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)*, pages 13–17, Montreal, QC, Canada, May 2019. IEEE.
- [18] J. Moreno-LeÓN, G. Robles, and M. RomÁN-González. Towards Data-Driven Learning Paths to Develop Computational Thinking with Scratch. *IEEE Transactions on Emerging Topics in Computing*, 8(1):193–205, Jan. 2020.
- [19] F. Obermüller, L. Bloch, L. Greifenstein, U. Heuer, and G. Fraser. Code Perfumes: Reporting Good Code to Encourage Learners. In *The 16th Workshop in Primary and Secondary Computing Education*, pages 1–10, Virtual Event Germany, Oct. 2021. ACM.
- [20] B. Paassen, J. McBroom, B. Jeffries, I. Koprinska, K. Yacef, et al. Mapping python programs to vectors using recursive neural encodings. *Journal of Educational Data Mining*, 13(3):1–35, 2021.
- [21] C. Piech, J. Huang, A. Nguyen, M. Phulsuksombati, M. Sahami, and L. Guibas. Learning program embeddings to propagate feedback on student code. In *International conference on machine Learning*, pages 1093–1102. PMLR, 2015.
- [22] M. Resnick, B. Silverman, Y. Kafai, J. Maloney, A. Monroy-Hernández, N. Rusk, E. Eastmond, K. Brennan, A. Millner, E. Rosenbaum, and J. Silver. Scratch: Programming for all. *Commun. ACM*, 52(11):60, Nov. 2009.
- [23] Y. Shi, T. Mao, T. Barnes, M. Chi, and T. W. Price. More with less: Exploring how to use deep learning effectively through semi-supervised learning for automatic bug detection in student code. In *In Proceedings of the 14th International Conference on Educational Data Mining (EDM) 2021*, 2021.
- [24] Y. Shi, K. Shah, W. Wang, S. Marwan, P. Penmetasa, and T. Price. Toward Semi-Automatic Misconception Discovery Using Code Embeddings. In *LAK21: 11th International Learning Analytics and Knowledge Conference*, pages 606–612, Irvine CA USA, Apr. 2021. ACM.
- [25] A. Swidan, F. Hermans, and M. Smit. Programming Misconceptions for School Students. In *Proceedings of*

- the 2018 ACM Conference on International Computing Education Research*, pages 151–159, Espoo Finland, Aug. 2018. ACM.
- [26] M. Talbot, K. Geldreich, J. Sommer, and P. Hubwieser. Re-use of programming patterns or problem solving?: Representation of scratch programs by TGraphs to support static code analysis. In *Proceedings of the 15th Workshop on Primary and Secondary Computing Education*, pages 1–10, Virtual Event Germany, Oct. 2020. ACM.
- [27] D. Vagavolu, K. C. Swarna, and S. Chimalakonda. A Mocktail of Source Code Representations. In *2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 1296–1300. IEEE, 2021.
- [28] J. Zhang, X. Wang, H. Zhang, H. Sun, K. Wang, and X. Liu. A novel neural source code representation based on abstract syntax tree. In *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*, pages 783–794. IEEE, 2019.

Investigating the effect of automated feedback on learning behavior in MOOCs for programming

Hagit Gabbay
School of Education
Tel Aviv University
hg.astd@gmail.com

Anat Cohen
School of Education
Tel Aviv University
anatco@tauex.tau.ac.il

ABSTRACT

The challenge of learning programming in a MOOC is twofold: acquiring programming skills and learning online, independently. Automated testing and feedback systems, often offered in programming courses, may scaffold MOOC learners by providing immediate feedback and unlimited re-submissions of code assignments. However, research still lacks empirical evidence of their effect on learning behavior of MOOC learners, with diverse backgrounds and goals. Addressing this gap, we investigated the connections between the use of automated feedback system and learning behavior measures, relevant for MOOCs: engagement, persistence and performance. Further, two subjective measures of success are examined: sense of learning and intention fulfilment. In an experimental design, we analyzed data of active learners in a Python programming MOOC (N=4652), comparing an experimental group provided with automated feedback with a control group that did not. In examining the effect of automated feedback, prior knowledge of programming and Python was considered. Empirical evidence was found for the relation between automated feedback usage and a higher engagement and better performance, as well as higher attendance in "active watchers" and "high-performed completers" clusters, obtained by cluster analysis. Learners reports on their experience with the automated feedback system supported these findings. Regarding the subjective measures of success, however, no difference was found between groups. Our study and the offered future research may contribute to the considerations regarding the integration of automated feedback in MOOCs for programming.

Keywords

automate feedback, MOOC for programming, learning behavior, prior knowledge, educational data mining, cluster analysis

1. INTRODUCTION

Massive Open Online Courses (MOOCs) for programming have the potential to teach programming to a broad and diverse audience [28]. The high demand for computer professionals and labor market needs have led to an abundance of courses, with large numbers of enrollees [25]. However, many learners struggle in these courses. Learning programming is challenging, but MOOC learners face additional difficulties, as they have to self-regulate their learning (SRL) and to cope with course content almost without the assistance of instructors [11]. The provision of feedback may assist

H. Gabbay and A. Cohen. Investigating the effect of automated feedback on learning behavior in MOOCs for programming. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 376–383, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853125>

learners with these challenges. Feedback is considered essential in online learning, both as formative assessment, promoting learning and increasing learner engagement and as an acceleration of the SRL process [16].

In programming courses, the practice of writing and running code is the basis for acquiring software developing skills [5, 48]. An immediate, detailed and accurate feedback makes practice more effective and may significantly improve learning [38, 40]. But in large scale courses, and especially in MOOCs, instructors cannot provide feedback on each submission [15]. Automated testing and feedback (ATF) systems address the need, allowing an unlimited number of learners and submissions [22]. Upon uploading a solution to the system, the learner receives immediate feedback and is given the option to resubmit a revised code, to complete the learning process. Feedback may address syntax errors, the correctness of results, the efficiency of the code and whether it fulfils instructions accurately [13, 48]. It can consist of basic feedback, including only correct/incorrect information or presenting the correct answer, or a detailed feedback, suggesting possible causes of error and hints for the solution [22, 44].

ATF systems were developed decades ago, and today there is a variety of tools and systems employing diverse technologies and methods for testing and generating feedback [9, 22, 37]. Previous research suggested that incorporating an ATF system into a programming course improves affective measures, such as satisfaction and sense of learning [4, 6]. The automated feedback is perceived by learners as enhancing learning as well as contributing to motivation and engagement [2, 32, 37]. Results regarding the system's impact on performance, however, were not conclusive (e.g. [7, 13, 14]).

Yet, most studies in the field of ATF have only focused on frontal courses, or online courses offered as part of an academic curriculum. Students in these courses have extensive interaction with the faculty, which enhances their learning [36, 42] and might "overshadow" the impact of automated feedback on learning outcome [15]. In MOOCs, ATF system may have a different effect. Furthermore, the diverse goals and intentions of MOOC learners may affect their learning behavior and performance [24, 29], and consequently the impact of automated feedback [34].

To gain the full potential of ATF in MOOCs, empirical research needs to be conducted in order to better understand the impact of automated feedback on learning behavior and outcomes. Yet, empirical research in this field is still lacking [1]. With the aim of addressing the gap, this study set out to investigate the connections between automated feedback usage and learners' learning behavior and outcomes, in MOOCs for programming.

It is now well established from a variety of studies that the relevant measures for learning behavior in MOOCs are engagement (measured through learning activity), persistence and performance [18,

20, 24, 51]. Additional success measures refer to learners' perception of achieved knowledge levels and intention fulfilment [17, 39]. Hence, the research questions we posed are as follows:

RQ1: What are the differences, if any, in engagement, persistence, and performance between ATF users and not-users?

RQ2: Is there any difference in learners' perceptions of achieved knowledge levels and intention fulfilment between ATF users and not-users?

Previous studies suggested that prior knowledge, related to the course content, may influence learning behavior and the impact of automated feedback [28, 33, 46]. Therefore, the investigated learning behavior measures were analysed with prior knowledge as a covariate.

Using a quantitative method and learning analytics approach, we compared data of two groups of learners in a Python programming MOOC incorporating ATF system: An experimental group who was given feedback on the code assignments by using the system, and a control group who did not. Collecting data from the course and system logs, cluster analysis was applied to identify and compare learning behavior patterns. Subjective success measures were compared based on learners' self-report data.

2. RELATED WORK

2.1 Measures of learning behavior in MOOCs

Previous research has established that given the heterogeneity of learners in MOOC, the appropriate measures of learning outcomes differ from those of formal education context [10, 24]. The most commonly used indicator to measure learning outcomes in MOOCs is learning engagement, measured by the number of lecture videos watched, the number of posts to forums, the number of quizzes taken, and the number of tasks completed [20, 52]. Another common measure is persistence, as measured by the learner's determination to complete tasks and the achieved degree of progress. The relevant measures are therefore the number of attempts to solve course exercises, the number of learning units that were studied (watching video or attempt to solve an exercise) as well as the most advanced unit that was studied [10, 20]. Learner's performance in the course is often measured by the scores on the exercises and assignments [18, 47].

Several studies applied cluster analysis to identify learning behavioral patterns and classified learner' groups by similar learning characteristics. A key study, establishing the main characteristics of MOOC learners, suggested four identified groups: completers (students who completed most assignments), auditing (students who did few-to-no assignments but engaged in watching videos), disengaging (students who did assignments early in the course, but later stopped participating), and sampling (students who watched videos only in the beginning of the course) [24]. A similar research described 3 groups of MOOC learners as: (1) active engaged (those who submit assignments and were actively involved in forums); (2) passive engaged (those who watch video or show passive involvement in forums); and (3) disengaged (learners whose activity decreased throughout the course) [41]. Recently, a study investigated learners' interactions with video, exercises and discussion forums, identifying seven patterns of learners' behavior: tasters, downloaders, disengagers, offline engagers, online engagers, and two patterns characterized by moderate and high social engagement [21]. The variables used to derive these patterns were the number of videos watched, in-video questions answered, exercises and assignments submitted, thread views and activity in the discussion forums.

Considering the suggested measures of learning behavior in MOOCs, in the current study we first applied cluster analysis to identified groups of learners who behaved similarly and then investigated the connections between behavior patterns and ATF usage.

2.2 Subjective measures of success in MOOCs

A variety of reasons motivate learners to enroll in MOOCs [29], with a variety of learning outcomes to expect [39]. Thus, it has been proposed that the success of learning in MOOCs should be evaluated through learner-centered measures. One suggested criterion is fulfillment of learner intentions [17]. In research conducting self-reported surveys, learners' intention fulfillment was found to be correlated with their engagement in course activities [39]. "Sense of learning" or "sense of achievement", representing the perceived increase of knowledge, is another criterion for successive learning, being in use in previous studies to measure learning success and outcomes in MOOCs [30, 49, 51].

2.3 The effect of ATF in MOOCs on learning

In the context of MOOCs for programming, only a few studies have examined the impact of automated feedback, provided on code assignment, on learning behavior and outcomes. Regarding the aforementioned affective measures, several studies noted that automated feedback was perceived by learners as improving performance, increasing engagement and having a positive impact on learning strategies [8, 27]. An interesting study suggested that learners who "committed" by formal registration to use an ATF system in MOOC for programming were more engaged in solving code assignments, in compared with those who used it only partially, without registration [12]. Recent study came up with similar results [43]. No effect of using the system was found, however, regarding performance and course completion rates.

Most studies on automated feedback propose advanced algorithms and new approaches to improve error detection and feedback accuracy, but do not evaluate its effectiveness as a learning tool [31, 45, 50]. Several research reports on future intention to evaluate the impact of the examined ATF system on learning outcomes, yet to be done [5, 25]. Others suggest factors to consider while choosing or developing ATF systems for MOOCs, but no empirical results are provided [48].

Overall, there seems to be some evidence to indicate that automated feedback has the potential to support learners and enhance learning success in MOOCs for programming. The findings of the current study contribute to empirically based knowledge in this area.

3. METHOD

3.1 Course and ATF system

To answer research questions, we conducted an experiment in MOOC to learn the Python programming language, offered on Edx-based platform for MOOCs. The course consists of nine learning units, each of which includes content videos with comprehension questions, closed exercises (such as multiple-choice questions), and code-writing assignments. Answering the closed exercises is followed by feedback (correct/incorrect and an explanation).

The ATF system integrated in the course is INGINIOUS - an open-source software, suitable for online programming courses, providing grades and textual feedback for code assignments (for more details on INGINIOUS, see [19]). The system was incorporated into the course as an external tool, and registration was necessary for access. It was configured to allow unlimited submission of solutions. The textual messages provided as part of the feedback were adapted to the code assignments, containing different levels of

feedback according to error-type (e.g. correct / incorrect, expected correct answer or more elaborated feedback), as classified by [44].

3.2 Experimental design

Using the cohort-mechanism embedded in Edx platform, we randomly divided the learners enrolled for the course into control group (control-g) and experimental group (atf-g). Learners in the experimental group gained access to the ATF system. Those who chose to use it uploaded solutions for the code assignments, received feedback, and then were able to resubmit revised solutions. In the current experimental set, however, it was not possible to get information about how the learners in the control group solved the code assignments.

3.3 Data resources and definitions of measures

Research questions were answered by gathering data from different sources and harvesting measures to be compared. Definitions of research variables are provided in this section.

- 1) Demographics and prior knowledge (PK) in programming and Python obtained by pre-course questionnaire. To avoid subjective assessment, PK was defined in a Boolean manner (there is / there is no PK). From learners' responses we derived three PK categories: None, Programming (other language), Python [3].
- 2) Learning behavior data, consists of engagement, persistence and performance, obtained from course event logs (table 1). As we do not have information regarding the number of code assignments solved by learners who did not use the ATF system (control group and learners in atf-g who chose not to use the system), it was not defined as a measure of activity.
- 3) Log files of the ATF system, which contain information on submitted solutions, were analyzed to assess the use of automated feedback (relevant only to the experimental group).
- 4) Subjective measures were collected from the "learning experience" questionnaire, completed by learners at the end of learning. "Sense of learning" refers to the learner's evaluation of the level of knowledge achieved at the end of learning process. Learners were asked to choose one of four statements, representing four levels of knowledge. Intention fulfillment defined as one of three values (Yes / Partially / No).
- 5) Learners in the experimental group were asked to answer two additional questions regarding their perception of how the use of the system affected their engagement (Likert scale 1-5): "The system contributed to the motivation to complete more tasks in the course", "The option to correct my solution and resubmit prompted me to make an effort for a higher score". The obtained responses were used as supportive data to the results of the comparison between the two research groups.

The research was conducted under the rules of ethics, while protecting privacy and maintaining the security of information, and in accordance with the approval of the university ethics committee.

3.4 Research population

The study was conducted in the second half of 2021. Following the screening of non-active enrollees and learners who did not provide information about prior knowledge, N=4652 learners were included for our final study population, 15.6% (724) of which in the experimental group (hereinafter: atf-g) and 84.4% (3928) in the control group (hereinafter: control-g). The imbalance between the research groups was caused by technical reasons, as the system was integrated for the first time during the course cycle in which the study was conducted. To ensure that none of this affected the results, all

Table 1 : Learning behavior calculated measures

	variable	Description
Engagement (activity)	watched video	Number of watched videos (0-29)
	watched units	Number of units in which at least one video was watched (0-9)
	active-watched ratio	Ratio between the number of videos in which the learner solved a comprehension question and the total number of videos watched (0-1)
	solved exercises	Number of closed exercises (CE) learner attempted (0-39)
	solved units	Number of units in which at least one exercise was attempted (0-9)
	mean attempts	Mean attempts per exercise
persistence	units touched	Number of units in which the learner watched a video or attempted an exercise (0-9)
	max unit touched	The most advanced unit in which the learner watched a video or attempted an exercise (1-9)
performance	grade	The mean score in closed exercises (0-10)

the analyses were repeated several times, comparing the experimental group (atf-g) with random groups drawn from the control group, of the same size as atf-g. Upon completion of this study, which was also a technical pilot, all learners in the following course cycles were able to enjoy the benefits of using the ATF system.

Participants were all Hebrew speakers, which is the language in which the course is taught, and were interested in learning Python. In terms of gender identification, learners' distribution was 73.5% male, 25.9% female, and 0.6% unidentified. The learners ranged in age from less than 11 to over 75, with the majority in the age at the range of 12-34 (79.3%).

According to self-reported prior knowledge, 30.4% of learners had prior programming skills but not Python (PK=Programming), 15.1% reported of Python (and programming) knowledge (PK=Python) and 54.5% had no prior knowledge relevant to course content (PK=None). Chi-square test indicated no significant difference between the experimental and control groups regarding the distribution of prior knowledge.

4. FINDINGS

This section presents the results aimed to answer research questions. To control the impact of PK on the effect of automated feedback as a covariate, the comparison of the two research groups in all tests was conducted separately at each level of PK. The findings are presented without regard to this covariate unless it is found to affect the results in a significant manner.

In regard to the imbalanced research groups, five repeats of the analyses with random subgroups from control-g, equal to the size of atf-g, yielded identical answers to research questions. Therefore,

we have chosen to present the comparison between the experimental group ($N_{\text{atf-g}} = 724$) and the entire control group ($N_{\text{control-g}} = 3928$).

Table 2: Descriptive statistics of learning behavior measures (Natf-g = 724, Ncontrol-g = 3928 M=mean, MD=median, SD=standard deviation)

variable	atf-g			Control-g		
	M	MD	SD	M	MD	SD
watched video	12.7	11	9.0	8.8	5	8.6
watched units	4.3	4	2.9	3.1	2	2.7
active-watched ratio	0.4	0.4	0.2	0.3	0.3	0.3
solved exercises	15.2	12	12.9	8.3	1	11.8
solved units	4.3	4	3.1	2.4	1	2.9
mean attempts	2.4	2.1	2.0	1.7	1.6	2.0
units touched	4.7	4	3.0	3.4	2	2.8
max unit touched	4.8	4	3.0	3.7	2	3.0
grade	0.8	1	0.3	0.5	0.8	0.5

4.1 RQ1: The connection between ATF usage, learning behavior and performance

4.1.1 Comparing learning behavior variables

The operational variables of learning behavior and performance (table 1) were calculated from data and compared between research groups. Mann-Whitney test was applied, as the homogeneity assumption required for the t-test was not met. In the experimental group, the mean and median values of all behavioral variables were higher, as illustrated in table 2 and figure 1. This difference was found to be significant at the $p < .001$ level (Mann-Whitney U ranged between 1050975 – 954295.5), with small-medium effect size, given by the rank biserial correlation (0.228 - 0.402).

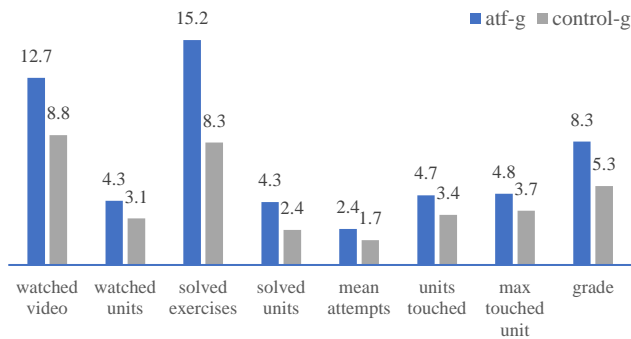


Figure 1: Mean values of learning behavior variables

Another way to measure persistence is to compare, per number of “touched” units, the percentage of learners from each group who reached this number. Chi-square test revealed significant difference between learners’ percentage of each group ($\chi^2(8) = 165.34$, $N = 4652$, $p < .001$). From three units above, the number of units “touched” by the experimental group is higher (see figure 2).

More specifically, it can be claimed that a higher percentage of learners from the experimental group completed the course, i.e. learned all the lessons. Further evidence of this fact comes from a finding that a higher percentage of learners from the experimental group studied units 7-9, the advanced units of the course. A significant difference was indicated between learners’ percentage from



Figure 2: Comparison of learners’ percentage per number of units “touched” (Natf-g = 724, Ncontrol-g = 3928)

each group who solved exercises in these units ($\chi^2_{\text{unit 7}}(2) = 100.00$, $\chi^2_{\text{unit 8}}(2) = 67.20$, $\chi^2_{\text{unit 9}}(2) = 34.42$, $p < .001$). Nevertheless, for learners with prior knowledge of Python, the difference between the groups regarding solving the exercises of the last unit (unit 9) was not statistically significant.

4.1.2 Cluster analysis of learning behavior variables

Among the variables representing learning behavior and performance, five “differentiating” variables were identified using PCA: active watched ratio, solved units, mean attempts, max unit touched and grade. k was assumed a priori to be between 4 and 6 so that the clusters would be distinct, but not too many. The elbow method plot and silhouette score were then used to identify the most fitting number of five clusters, explaining 72.7% of variance in data ($R^2 = 0.727$). Table 3 summarizes the clusters obtained and the mean values of the differentiating variables within each cluster. Following the behavior patterns characterizing each cluster, they were named as follows: (1) “Touched and left”: those who log in but showed almost no engagement with course content. This pattern, which represents learners who actually dropped out shortly after they started, was the most frequent. (2) “Completers, high performers”: learners with highest performance and completing rates, while only moderate consumption of content. This pattern was the second in number of learners (3) “Active-watchers”: those who watched video actively, answering in-video comprehension questions. (4) “Good starter, mean progress”: those who solved correctly few exercises, mainly of the first one or two units, but had no intention to complete the course. (5) “Trail-error solvers”: those who try to solve few exercises, with many attempts, low success and no progress. This was the least desirable behavior pattern.

Examining the presence of learners from each group within each of these clusters, presented in figure 3, revealed relatively high percentage of atf-g learners in clusters 2,3 (42.40%, 24.17%, respectively) and higher percentage of the control-g learners in cluster 1 (41.42%). Chi-square test indicated a statistically significant difference between atf-g and control-g ($\chi^2(4) = 277.208$, $N = 4652$, $p < .001$) over all levels of PK. Yet, cluster 2 was found to be significantly different from the other four clusters in terms of PK, as determined by one-way ANOVA ($F(4,4647) = 32.664$, $p < .001$)

and Tukey’s HSD test for multiple comparisons ($p < .001$ for all the comparisons of cluster 2 and other clusters).

Table 3: Cluster characteristics and mean value of learning behavior variables (In bold: characterizing attribute of each cluster)

Cluster	1	2	3	4	5
Size	1707	1161	897	756	131
percentage of N=4652	36.91%	25.10%	19.39%	16.35%	2.83%
Active-watched ratio	0.142	0.428	0.604	0.15	0.329
Solved units	0.164	7.205	2.557	1.655	1.473
Mean attempts	0.179	2.556	2.338	2.319	9.152
Max unit touched	2.17	7.999	3.198	2.298	2.183
Grade	< 1	95.1	87.4	95.1	67.4

4.1.3 Learners’ perception of ATF effects

Two questions about the effect of using the system were answered by 126 learners. In their responses, learners indicated they believed that using the ATF system affected their engagement and performance. The majority of respondents agreed with the statements that the option to correct and resubmit prompted them to make an effort for a higher score (87%) and using the ATF system motivated them to be more engaged in solving course exercises (81%). PK level had no impact on learners’ perceptions.

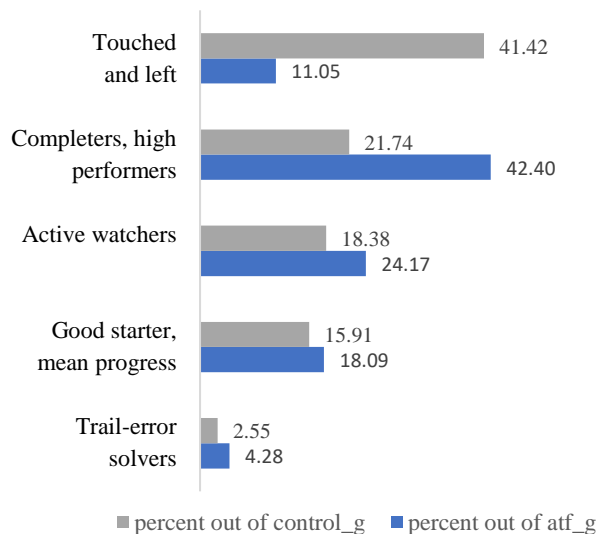


Figure 3: The percentage of learners from each group within each cluster

RQ2: Connections between automated feedback and success measures

Of the study population, 401 learners (9.87%) completed the “learning experience” questionnaire. Among the respondents, 32.27% (126) belong to the experimental group and 67.73% (275) to the control group. As can be seen in table 4, the majority of respondents described their achieved level of learning as “Can write basic code, needs more supported practice” (49.63%), followed by “Can write complex code and practice independently” (35.94%). Only 5.8% felt confident to start working in Python. To the question of intention fulfilments, however, the most frequent answer was “Yes” (81.91%).

Table 4: Subjective measures, learning experience questionnaire (N=401)

Sense of learning	
Understands the concepts, can't write code independently	8.56%
Can write basic code, needs more supported practice	49.63%
Can write complex code and practice independently	35.94%
Can work in Python	5.87%
Intention fulfilment	
No	0.98%
Partially	17.12%
Yes	81.91%

Applying the chi-square test to compare between research groups, no significant difference was found in all levels of PK. Notably, an expected significant dependency was found between PK and sense of learning ($X^2(6) = 67.73, N = 401, p < .001$), as prior knowledge led to higher assessment of the achieved level of knowledge.

5. DISCUSSION

With respect to the first research question, findings demonstrate connections between ATF system usage and learning behavior, as learners in the experimental group were more engaged with the course material and completed it at a higher rate. These results are in line with previous studies, suggesting that feedback (in general) enhances learners’ engagement and persistence in online courses (e.g. [16, 20]) as well as in programming courses, not necessarily be offered online (e.g. [12]). Notably, in the current study all learners received feedback on the closed exercises, and the difference between groups was due to the additional feedback provided to the experimental group for code assignments. Therefore, it should be assumed that feedback on code assignments in MOOCs for programming is of utmost importance. Our findings, however, are contrary to those of [43] who did not observe a connection between feedback and learners’ engagement. A possible explanation for this might be differences in feedback and course characteristics, which affect feedback effectiveness [33].

The current study indicates a connection between automated feedback and learners’ performance and suggests positive trend of higher grades of the ATF users. Previous studies have not conclusively established this connection (e.g. [13, 14]). Nevertheless, due to our experimental design, we defined performance by closed exercises score and not by programming abilities (e.g. grades of code

assignments). As such, the evidence of higher performance of atg in this case may indicate in fact a deeper understanding of programming principles [26].

Our clustering of learners based on learning behavior variables is similar to behavior patterns classified in previous studies [21, 24, 41], even though the number of clusters we selected is different. Learners of the experimental group were more “present” in the clusters described as “completers, high performers” and “Active-watchers” characterized by patterns identified as related to success in MOOCs [23]. The cluster analysis results support and expand the findings obtained for each variable as stand alone.

Interestingly, most of the analyses we conducted did not reveal any effect of prior knowledge on the connection between learning behavior and automated feedback. These findings do not support previous study, suggesting that prior knowledge affects the effectiveness of feedback in online learning environments [35]. Further research, with higher resolution determination of prior knowledge level, may shed light on this issue. One exceptional of our results is the case in Unit 9, where similar percentages of the Python experienced learners of two groups solved the closed exercises. It may be that the prior Python knowledge obscured the differences between groups, as experienced learners wanted to test their knowledge in this very specific unit, which is the most advanced one.

In support of the findings based on (objective) log files, the perceptions of the learners in the experimental group suggest that the automated feedback they provided heightened their motivation to be more engaged in solving exercises and encouraged them to score higher. This attitude is similar to learners’ perception towards automated feedback reported in previous studies, in the context of in-class programming courses (e.g. [2, 32]). With regards to affective measures, therefore, automated feedback is perceived beneficial in both frontal and online learning environments.

With respect to the second research question, however, finding of the current study do not indicate a connection between the use of the ATF system and the subjective measures of success - intention fulfillment and sense of learning. These results differ from previous studies which have suggested that intention fulfillment is correlated with engagement in solving exercises in MOOC [39]. A two-way effect may have been created here: On one hand, the automated feedback may have led learners in the experimental group to more comprehensive learning [38] and on the other, they were more aware of errors and incorrect solutions, leading to a lower assessment of their abilities in Python.

6. CONCLUSIONS AND FUTURE WORK

Overall, the results of this study indicate a connection between the use of the ATF system, providing automated feedback, and learning behavior. Furthermore, the findings suggest that the automated feedback enhances the learners’ engagement and persistence in the course as well as their performance. Nevertheless, we must be cautious in this context, and further research is needed to examine the effects of feedback on learning behavior (e.g. examining a “directional” connection). This is primarily due to the finding that the sense of learning and intention fulfilment were not affected by the use of ATF, suggesting the effect of feedback is likely to be complex and non-uniform within different facets of learning outcomes. However, the inability to obtain an objective assessment of the level of knowledge at the end of the learning is one of the current study limitations. The result of a final exam, for example, may reveal differences between research groups that are not apparent in self-reported evaluation. Nonetheless, there is no mandatory assessment in a MOOC due to its nature.

A further limitation is that the experimental design prevented a comparison of the research groups in regard to solving code assignments, which in fact is the subject of feedback. Future research be undertaken with a setup allowing the comparison of these data as well, might bring additional insight into the effect of automated feedback.

Lastly, the feedback provided by the ATF system was referred to in the current study as “black box”. In light of the concept proposed by Narciss [33], which links the characteristics of feedback to its effectiveness, further experimental research is needed to analyze the effects of feedback characteristics (e.g. the structure of textual message) on learning in MOOCs for programming. Expanding empirical research knowledge regarding the impact of automated feedback on learning may contribute to the effective integration of ATF systems and thus promoting learners’ acquisition of programming skills and achievement of learning goals.

7. ACKNOWLEDGMENTS

Our thanks to the Azrieli foundation for the award of a generous Azrieli Fellowship, which allowed this research. We are also indebted to our anonymous reviewers and the EDM program chairs for their valuable feedback.

8. REFERENCES

- [1] Ahmed, U.Z., Srivastava, N., Sindhgatta, R. and Karkare, A. 2020. Characterizing the pedagogical benefits of adaptive feedback for compilation errors by novice programmers. *Proceedings - International Conference on Software Engineering* (2020), 139–150.
- [2] Annamaa, A., Suviste, R. and Vene, V. 2017. Comparing different styles of automated feedback for programming exercises. *ACM International Conference Proceeding Series* (2017), 183–184.
- [3] Bajwa, A., Bell, A., Hemberg, E. and O’Reilly, U.M. 2019. Student code trajectories in an introductory programming MOOC. *Proceedings of the 6th 2019 ACM Conference on Learning at Scale, L@S 2019* (New York, NY, USA, Jun. 2019), 1–4.
- [4] Benotti, L., Aloï, F., Bulgarelli, F. and Gomez, M.J. 2018. The effect of a web-based coding tool with automatic feedback on students’ performance and perceptions. *SIGCSE 2018 - Proceedings of the 49th ACM Technical Symposium on Computer Science Education* (2018), 2–7.
- [5] Bey, A., Jermann, P. and Dillenbourg, P. 2018. A comparison between two automatic assessment approaches for programming: An empirical study on MOOCs. *Educational Technology and Society*. 21, 2 (2018), 259–272. DOI:<https://doi.org/10.2307/26388406>.
- [6] Cai, Y.Z. and Tsai, M.H. 2019. Improving Programming Education Quality with Automatic Grading System. *International Conference on Innovative Technologies and Learning* (Dec. 2019), 207–215.
- [7] Cavalcanti, A.P., Barbosa, A., Carvalho, R., Freitas, F., Tsai, Y.-S., Gašević, D. and Mello, R.F. 2021. Automatic feedback in online learning environments: A systematic literature review. *Computers and Education: Artificial Intelligence*. 2, (Jan. 2021), 100027. DOI:<https://doi.org/10.1016/J.CAEAI.2021.100027>.
- [8] Chan, M.M., De La Roca, M., Alario-Hoyos, C., Plata, R.B.,

- Medina, J.A. and Rizzardini, R.H. 2017. MOOCMaker-2017 Perceived Usefulness and Motivation Students Towards the use of a Cloud-based Tool to Support the Learning Process in a Java MOOC. *International Conference MOOC-MAKER* (2017), 73–82.
- [9] Combéfis, S. 2022. Automated Code Assessment for Education : Review , Classification and Perspectives on Techniques and Tools. *Software* 2022. 1, (2022), 3–30. DOI:<https://doi.org/https://doi.org/10.3390/software1010002>.
- [10] Evans, B.J., Baker, R.B. and Dee, T.S. 2016. Persistence Patterns in Massive Open Online Courses (MOOCs). <http://dx.doi.org/10.1080/00221546.2016.11777400>. 87, 2 (Mar. 2016), 206–242. DOI:<https://doi.org/10.1080/00221546.2016.11777400>.
- [11] Feklistova, L., Luik, P. and Lepp, M. 2020. Clusters of programming exercises difficulties resolvers in a MOOC. *Proceedings of the European Conference on e-Learning, ECEL*. 2020-October, (2020), 563–569. DOI:<https://doi.org/10.34190/EEL.20.125>.
- [12] Gallego-Romero, J.M., Alario-Hoyos, C., Estévez-Ayres, I. and Delgado Kloos, C. 2020. Analyzing learners' engagement and behavior in MOOCs on programming with the Codeboard IDE. *Educational Technology Research and Development*. 68, 5 (Oct. 2020), 2505–2528. DOI:<https://doi.org/10.1007/s11423-020-09773-6>.
- [13] Gordillo, A. 2019. Effect of an Instructor-Centered Tool for Automatic Assessment of Programming Assignments on Students' Perceptions and Performance. *Sustainability*. 11, 20 (Oct. 2019), 5568. DOI:<https://doi.org/10.3390/su11205568>.
- [14] Gusukuma, L., Bart, A.C., Kafura, D. and Ernst, J. 2018. Misconception-driven feedback: Results from an experimental study. *ICER 2018 - Proceedings of the 2018 ACM Conference on International Computing Education Research* (New York, New York, USA, Aug. 2018), 160–168.
- [15] Hao, Q., Wilson, J.P., Ottaway, C., Iriumi, N., Arakawa, K. and Smith, D.H. 2019. Investigating the essential of meaningful automated formative feedback for programming assignments. *Proceedings of IEEE Symposium on Visual Languages and Human-Centric Computing, VL/HCC* (Oct. 2019), 151–155.
- [16] Hattie, J. and Timperley, H. 2007. The power of feedback. *Review of Educational Research*. 77, 1 (2007), 81–112. DOI:<https://doi.org/10.3102/003465430298487>.
- [17] Henderikx, M., Kreijns, K., Education, M.K.-D. and 2017, undefined 2017. Refining success and dropout in massive open online courses based on the intention–behavior gap. *Taylor & Francis* 3, 38. (Sep. 2017), 353–368. DOI:<https://doi.org/10.1080/01587919.2017.1369006>.
- [18] Hew, K.F. 2016. Promoting engagement in online courses: What strategies can we learn from three highly rated MOOCs. *British Journal of Educational Technology*. 47, 2 (Mar. 2016), 320–341. DOI:<https://doi.org/10.1111/bjet.12235>.
- [19] INGIInious [software] 2014. <https://github.com/UCL-INGI/INGIInious>.
- [20] Jung, Y. and Lee, J. 2018. Learning engagement and persistence in massive open online courses (MOOCs). *Computers and Education*. 122, (Jul. 2018), 9–22. DOI:<https://doi.org/10.1016/j.compedu.2018.02.013>.
- [21] Kahan, T., Soffer, T. and Nachmias, R. 2017. Types of participant behavior in a massive open online course. *IRRODL* 18–1, (2017) 6, 18. DOI:<https://doi.org/10.19173/irrodl.v18i6.3087>.
- [22] Keuning, H., Jeurung, J. and Heeren, B. 2018. A systematic literature review of automated feedback generation for programming exercises. *ACM Transactions on Computing Education*. 19, 1 (2018), 1–43. DOI:<https://doi.org/10.1145/3231711>.
- [23] Khalil, M. and Ebner, M. 2017. Clustering patterns of engagement in Massive Open Online Courses (MOOCs): the use of learning analytics to reveal student categories. *Journal of Computing in Higher Education*. 29, 1 (Apr. 2017), 114–132. DOI:<https://doi.org/10.1007/S12528-016-9126-9/FIGURES/7>.
- [24] Kizilcec, R.F., Piech, C. and Schneider, E. 2013. Deconstructing disengagement: Analyzing learner subpopulations in massive open online courses. *ACM International Conference Proceeding Series*. (2013), 170–179. DOI:<https://doi.org/10.1145/2460296.2460330>.
- [25] Krugel, J., Hubwieser, P., Goedicke, M., Striewe, M., Talbot, M., Olbricht, C., Schypula, M. and Zettler, S. 2020. Automated Measurement of Competencies and Generation of Feedback in Object-Oriented Programming Courses. *2020 IEEE Global Engineering Education Conference (EDUCON)*, 336–329, (2020).
- [26] Krusche, S. and Seitz, A. 2018. ArTEMiS - An Automatic Assessment Management System for Interactive Learning. *Proceedings of the 49th ACM Technical Symposium on Computer Science Education* (New York, NY, USA, 2018), 284–289.
- [27] Krusche, S. and Seitz, A. 2019. Increasing the Interactivity in Software Engineering MOOCs-A Case Study. *Proceedings of the 52nd Hawaii International Conference on System Sciences* (2019), 7592–7601.
- [28] Luik, P., Feklistova, L., Lepp, M., Tönisson, E., Suviste, R., Gaiduk, M., Säde, M. and Palts, T. 2019. Participants and completers in programming MOOCs. *Education and Information Technologies*. 24, 6 (Nov. 2019), 3689–3706. DOI:<https://doi.org/10.1007/s10639-019-09954-8>.
- [29] Luik, P., Lepp, M., Feklistova, L., Säde, M., Rõõm, M., Palts, T., Suviste, R. and Tönisson, E. 2020. Programming MOOCs—different learners and different motivation. *International Journal of Lifelong Education*. 39, 3 (May 2020), 305–318. DOI:<https://doi.org/10.1080/02601370.2020.1780329>.
- [30] Magen-Nagar, N. and Cohen, L. 2017. Learning strategies as a mediator for motivation and a sense of achievement among students who study in MOOCs. *Education and Information Technologies*. 22, 3 (May 2017), 1271–1290. DOI:<https://doi.org/10.1007/S10639-016-9492-Y/FIGURES/3>.
- [31] Marin, V.J., Pereira, T., Sridharan, S. and Rivero, C.R. 2017.

- Automated personalized feedback in introductory Java programming MOOCs. *Proceedings - International Conference on Data Engineering*. (2017), 1259–1270. DOI:<https://doi.org/10.1109/ICDE.2017.169>.
- [32] Marwan, S., Fisk, S., Price, T.W., Barnes, T. and Gao, G. 2020. Adaptive Immediate Feedback Can Improve Novice Programming Engagement and Intention to Persist in Computer Science. *The 2020 ACM Conference on International Computing Education Research*–194 ,(2020) 203.
- [33] Narciss, S. 2013. Designing and evaluating tutoring feedback strategies for digital learning environments on the basis of the interactive tutoring feedback model. *Digital Education Review*. 23, 1 (2013), 7–26. DOI:<https://doi.org/10.1344/der.2013.23.7-26>.
- [34] Narciss, S. 2008. Feedback strategies for interactive learning tasks. *Handbook of research on educational communications and technology*. J.M. Spector, M.D. Merrill, J. Van Merriënboer, and M.P. Driscoll, eds. Lawrence Erlbaum Associates, Mahaw, NJ. 125–144.
- [35] Narciss, S., Sosnovsky, S., Schnaubert, L., Andrès, E., Eichelmann, A., Gogvadze, G. and Melis, E. 2014. Exploring feedback and student characteristics relevant for personalizing feedback strategies. *Computers and Education*. 71, (Feb. 2014), 56–76. DOI:<https://doi.org/10.1016/j.compedu.2013.09.011>.
- [36] Nicol, D.J. and Macfarlane-Dick, D. 2006. Formative assessment and self-regulated learning: a model and seven principles of good feedback practice. *Studies in Higher Education*. 31, 2 (Apr. 2006), 199–218. DOI:<https://doi.org/10.1080/03075070600572090>.
- [37] Pettit, R. and Prather, J. 2017. Automated assessment tools: too many cooks, not enough collaboration. *Journal of Computing Sciences in Colleges*. 32, 4 (2017), 113–121.
- [38] Pieterse, V. 2013. Automated Assessment of Programming Assignments. *3rd Computer Science Education Research Conference on Computer Science Education Research*. 3, April (2013), 45–56. DOI:<https://doi.org/http://dx.doi.org/10.1145/1559755.1559763>.
- [39] Rabin, E., Kalman, Y.M. and Kalz, M. 2019. An empirical investigation of the antecedents of learner-centered outcome measures in MOOCs. *International Journal of Educational Technology in Higher Education*. 16, 1 (Dec. 2019), 1–20. DOI:<https://doi.org/10.1186/S41239-019-0144-3/TABLES/4>.
- [40] Rafique, W., Dou, W., Hussain, K. and Ahmed, K. 2020. Factors influencing programming expertise in a web-based e-learning paradigm. *Online Learning Journal*. 24, 1 (2020), 162–181. DOI:<https://doi.org/10.24059/olj.v24i1.1956>.
- [41] Ramesh, A., Goldwasser, D., Huang, B., Iii, H.D. and Getoor, L. 2013. Modeling learner engagement in MOOCs using probabilistic soft logic. *NIPS workshop on data driven education* (2013), Vol. 21, 62.
- [42] Restrepo-Calle, F., Ramírez Echeverry, J.J. and González, F.A. 2019. Continuous assessment in a computer programming course supported by a software tool. *Computer Applications in Engineering Education*. 27, 1 (Jan. 2019), 80–89. DOI:<https://doi.org/10.1002/cae.22058>.
- [43] Serth, S., Teusner, R. and Meinel, C. 2021. Impact of Contextual Tips for Auto-Gradable Programming Exercises in MOOCs. *Proceedings of the Eighth ACM Conference on Learning @ Scale* (New York, NY, USA, 2021), 307–310.
- [44] Shute, V.J. 2008. Focus on Formative Feedback. *Review of Educational Research*. 78, 1 (Mar. 2008), 153–189. DOI:<https://doi.org/10.3102/0034654307313795>.
- [45] Singh, R., Gulwani, S. and Solar-Lezama, A. 2013. Automated feedback generation for introductory programming assignments. *Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI)*. (2013), 15–26. DOI:<https://doi.org/10.1145/2462156.2462195>.
- [46] Smits, M.H.S.B., Boon, J., Sluijsmans, D.M.A. and van Gog, T. 2008. Content and timing of feedback in a web-based learning environment: effects on learning as a function of prior knowledge. *Interactive Learning Environments*. 16, 2 (Aug. 2008), 183–193. DOI:<https://doi.org/10.1080/10494820701365952>.
- [47] Soffer, T. and Cohen, A. 2019. Students’ engagement characteristics predict success and completion of online courses. *Journal of Computer Assisted Learning*. 35, 3 (2019), 378–389. DOI:<https://doi.org/10.1111/jcal.12340>.
- [48] Staubitz, T., Klement, H., Renz, J., Teusner, R. and Meinel, C. 2015. Towards practical programming exercises and automated assessment in Massive Open Online Courses. *Proceedings of 2015 IEEE International Conference on Teaching, Assessment and Learning for Engineering, TALE 2015* (2015), 23–30.
- [49] Stich, A.E. and Reeves, T.D. 2017. Massive open online courses and underserved students in the United States. *The Internet and Higher Education*. 32, (Jan. 2017), 58–71. DOI:<https://doi.org/10.1016/J.IHEDUC.2016.09.001>.
- [50] Wang, K., Lin, B., Rettig, B., Pardi, P. and Singh, R. 2017. Data-driven feedback generator for online programming courses. *The Fourth (2017) ACM Conference on Learning@ Scale260–257* ,(2017) .
- [51] Wei, X., Saab, N. and Admiraal, W. 2021. Assessment of cognitive, behavioral, and affective learning outcomes in massive open online courses: A systematic literature review. *Computers & Education*. 163, (Apr. 2021), 104097. DOI:<https://doi.org/10.1016/J.COMPEDU.2020.104097>.
- [52] Xiong, Y., Li, H., Kornhaber, M.L., Suen, H.K., Pursel, B. and Goins, D.D. 2015. Examining the Relations among Student Motivation, Engagement, and Retention in a MOOC: A Structural Equation Modeling Approach. *Global Education Review*. 2, 3 (2015), 23–33.

Data-driven goal setting: Searching optimal badges in the decision forest

Julian Langenhagen
Goethe University Frankfurt
langenhagen@econ.uni-frankfurt.de

ABSTRACT

Although badges are among the most-used game elements in gamified education, studies about their optimal features to motivate learning are scarce. How should a badge be designed to represent an incentive for a specific goal like optimal exam preparation? This study examines usage data of a higher education learning app to determine whether the used badges have the intended motivational effect. The preliminary results suggest that the badges that were initially implemented in the app have the intended effect in most cases, but the stages of the multi-level badges could be optimized. The methodological framework used in this study can be transferred to usage data of other similar learning tools. With the help of easy-to-interpret outputs of decision trees, researchers and practitioners alike can work towards an optimal badge design.

Keywords

Gamification, Education, Badges, Decision Trees

1. INTRODUCTION

Badges are among the most used elements in game-based learning [1, 18, 14]. Yet, little is known about their optimal design in the educational domain. I define a design in this context as optimal if a badge can only be earned by following a learning strategy that leads to improved learning outcomes. The inspiration for the design of badges in game-based learning often comes from entertainment games [3, 22]. However, the goals of such games are usually different from those of learning tools used in school or higher education [20]. This difference also applies to commercial learning apps such as Duolingo. In Duolingo, for example, it is possible to earn a badge called “Weekend Warrior” if you use the app on a Saturday or Sunday and a badge called “Photogenic” if you upload a profile picture¹. These two badges are presumably intended to maximize usage time or identification with the app and are not necessarily aimed

¹<https://duolingo.fandom.com/wiki/Achievements>

J. Langenhagen. Data-driven goal setting: Searching optimal badges in the decision forest. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 384–390, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853163>

at an optimal learning strategy. Since designers of game-based learning tools may use such commercial apps as a blueprint, they run the risk of copying badges that motivate non-optimal learning strategies. This study should help educators that use badges in their learning tools to evaluate whether their chosen design motivates learning strategies that lead to better learning outcomes.

Empirical studies that investigate optimal badge design in learning contexts are rare. Easley and Ghosh [9] analyzed optimal badge design with a game-theoretic approach from a general but not an educational point of view. Antonaci et al. [1] link the effects of different game elements on learning outcomes in their literature review to investigate the question of which game elements cause which effects. They find that some game elements can have either a positive or a negative effect depending on the context (e.g., specific setting, other game elements used and the personality of the students). This result is also confirmed by recently conducted meta-studies [2, 15, 24]. They all report an overall positive significant small to medium effect size of game-based learning on learning outcomes. However, it is still unclear which factors contribute most to the success or failure of game-based education. In addition, the meta-studies above shed light only on the overall effects of combinations of different game elements. A more in-depth analysis of different badge designs (or other single game elements) does not exist yet. Facey-Shaw et al. [12] provide an overview of different badge designs and their functions in various educational applications. Still, they do not link the different designs to the achieved effects on the learning outcomes. The present study will contribute to this research gap.

The data set used in this study consists of usage data from a gamified learning app that has been in use for four semesters in an accounting lecture at a large European university. Among other game elements, the app uses badges to motivate the students to learn. The required achievements to earn the badges include specific amounts of consecutive usage days or answered questions. The corresponding goals were developed based on learning theories and practical experiences from past lectures but without any empirical validation. The goal of this study is to answer the question of whether students who aligned their learning strategy to the given badge goals performed better on the exam. Based on the results, the basic features but also the levels of the badges can be discussed. For example, the badge awarded for the amount of answered questions in the data set has

three levels. The highest level is earned when a student answers 1,000 questions. Unless a large proportion of the top students answered that many questions in the app, you could conclude that the threshold might be too high. If the level is too high, earning that badge could even be a predictor for a bad performance in the exam, i.e., students who answer too many questions might overestimate the app as a learning tool and neglect other learning materials. The following two research questions will be answered in this study to contribute to the discussed research gap:

RQ1: Do the badges that were initially implemented in the gamified learning app incentivize a learning strategy that leads to good exam results?

RQ2: Are the levels of the multi-level badges optimally set?

2. RESEARCH SETTING

2.1 Lecture and App

The research subject of this study is data from a gamified learning app that was used in an undergraduate accounting course at a large public university in Europe. The course is usually taken by approximately 600 students per semester (see Table 1) and consists of a weekly lecture and bi-weekly tutorials and small group exercises. Attendance at the classes is voluntary. The learning materials consist of a script, a collection of exercises and a trial exam. In addition, a textbook and the corresponding exercise book are recommended for further study. Apart from the written exam at the end of the semester, there is no other formal assessment. In evaluations of past semesters, students often commented that there was no contemporary way to engage with the subject matter in addition the aforementioned traditional learning materials. As a result, the responsible chair applied for funding and planned to develop a smartphone app with gamification elements.

The launch of the app took place in the summer semester of 2019. Like attendance at the classes, the use of the app is voluntary and no formal advantages for the exam can be gained by using the app. The app is widely accessible, as in addition to an app version in Google Play Store and Apple App Store, it is also available as a web version that can be accessed via the browser. Over 550 questions were developed for the quiz app, covering all nine chapters of the course. There are four question types in the app: single and multiple-choice, as well as sorting and cloze text tasks. Students can use the app in three different modes: Chapter Mode, Random Mode, and Weekly Challenge. In Chapter Mode, students can choose to be shown only questions from a specific chapter. When they master a chapter, the next chapter is unlocked. The current status in a chapter is visualized by a progress bar. The corresponding progress is measured by the so-called Skill Level Indicator (SLI), which was designed based on Ebbinghaus's forgetting curve [10]. When a question is answered correctly, the SLI increases to 100%. The result is that the corresponding question is no longer displayed for the time being. However, over time, the SLI decreases again to indicate that the content of the question may have been forgotten. After a certain duration (which becomes increasingly longer after answering the same question several times), the question is displayed again to check whether it is still correctly answered. In the second

mode, Random Mode, questions are randomly drawn from the pool that have already been unlocked by the student. In the third mode, the Weekly Challenge, students can answer 25 randomly selected questions once a week from chapters that have already been covered in the lecture. They can then compare their performance with that of fellow students on a leaderboard. The more questions students answer, the more learning points they earn in the app and level up accordingly. Moreover students can earn the before mentioned badges for certain achievements (see Figure 2). A detailed description of the initial badge set is outlined in Section 2.4 and more information about the app in general and its usage can be found in [19].

2.2 Data Sources

The collected data in this study consists of three different sources over four semesters (see Table 1): the exam results of every enrolled student, the app usage data and responses from a survey conducted each semester.

Table 1: Data Sources and Sample Sizes

	SS19	WS19	SS20	WS20
App	559	595	447	546
Exam	575	648	616	644
Survey	127	108	114	167
App+Exam	230	243	190	190

The collected usage data from the app consists of various details, including the exact time a question was answered, the answer given, and whether it was correct or incorrect. The functions and design of the app did not change in any of the four semesters considered in this study. Since no laboratory conditions prevail in the present setting, it was important to keep as many parameters constant as possible [5, 11]. In addition to the app version, the instructors, the learning materials, and the type of exam were not changed over the entire observation period. In each semester, the exam consisted mainly of arithmetic problems and contained three major tasks, each worth 30 points, which were divided into smaller sub-tasks. Even though the style of the exam remained identical over the semesters, it cannot be guaranteed that the level of difficulty remains the same between semesters. This circumstance is considered in the evaluation and assignment of grades, i.e., a specific number of points can result in a different grade in different semesters. Therefore, the analysis in this study uses grades rather than points as the dependent variable to control for the effect of varying difficulty. Grades in this setting range from 1.0 (very good) to 5.0 (failed).

One limitation regarding the analysis of the data is that I need the students' matriculation numbers to be able to connect the data points from the app and the exam. Due to data protection rules, I was obliged to ask the students for their matriculation number in the app on a voluntary basis. As not every student provided his or her matriculation number, I am only able to connect the data of the two data sources for a limited number of students (see Table 1). Therefore, I cannot analyze the app's influence on every exam result because I do not know for every student in the exam score list to which extent they used the app. To address this issue, a later section of this paper will examine

whether students who gave consent for their data to be used for research purposes differ significantly from students who withheld consent.

The third data source is a survey that was conducted every semester. Although the average response rate of the survey (26%) is comparable with similar prior studies [25, 17, 26], I decided not to connect the survey answers to the other two data sources for the present study. To be able to connect the survey answers, the students also needed to submit their matriculation number with the survey answers. As only a small fraction followed this request, the sample size with information from all three data sources would be significantly smaller than the sample size with data only from the app and the exam (see Table 1). However, Table 2 shows with data from the survey that the four cohorts did not differ significantly in the major demographics and can therefore be considered comparable for this study.

Table 2: Demographics of the Four Cohorts

	SS19	WS19	SS20	WS20
Male	59.05%	53.70%	47.30%	49.18%
Female	40.95%	46.30%	52.70%	50.82%
∅ Age	21.82	21.10	21.81	21.24
∅ Semester	3.73	3.27	3.61	3.41

2.3 Methodology

To analyze the badges implemented initially, the badges and the corresponding learning metrics are discussed below in a first step. In a second step, to answer the first research question, I will analyze whether students who followed the learning strategy motivated by the badges performed better in the exam by comparing the average grades. To answer the second research question, a decision tree is built to illustrate the influence of certain limits of the previously mentioned learning metrics on the average grades. The decision tree method is used because its results are easy to interpret and explain. This advantage is significant if the results and the corresponding analysis framework are to be used in practice. For this, especially the graphical representation method is very suitable, especially in comparison to a traditional regression output. The disadvantages of the method are the lower predictive accuracy (e.g., compared to random forests) and the fact that they are less robust than other approaches [16]. However, from my point of view, the advantages outweigh the disadvantages for the chosen purpose. All analyses for this study were conducted in R (4.1.1) and to create the decision tree the packages `rpart` (4.1-15) and `rpart.plot` (3.1.0) were used. The resulting flow chart of a decision tree can be used to evaluate and adjust the limits of the multi-level badges to increase the motivational effect if necessary. In practice, the optimized badge set would then be used for another term and afterwards the resulting usage data would again be evaluated with regards to the target variable. This general iterative process can be summarized as outlined in Figure 1.

2.4 Initial Badge Design

The initial badge set included badges in six categories (see Figure 2). The badges can be divided into single-level badges (one learning metric results in one single badge) and multi-level badges (one learning metric results in multiple badges

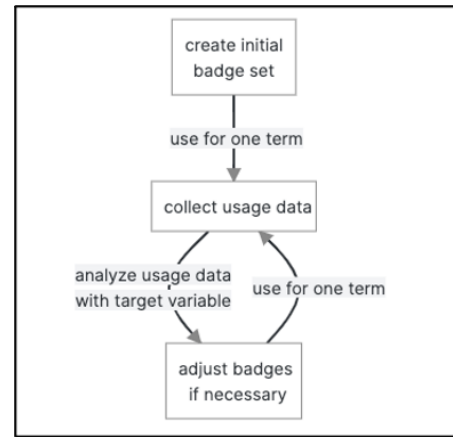


Figure 1: Iterative Optimization Process

divided into different stages). The first badge was awarded for students entering their matriculation number to share their usage data for research purposes. Since this is not a learning strategy or a measure of learning success, this badge will not be discussed further in this study. The remaining five badge categories and the corresponding learning metrics are discussed in the following sections.

2.4.1 Unique Questions

The second badge in Figure 2 is awarded if a student answered every question in the database correctly once. The influence of this badge is evaluated with the learning metric “Unique Questions”. This metric measures how many different questions a user has answered. As explained in Section 2.1, the SLI ensures that a question that has been answered correctly is only displayed again after a certain time. Nevertheless, each question can in principle be answered multiple times, which is why “Unique Questions” provides valuable information. The further a user has worked through the chapters, the higher this metric will be. If a student uses the app regularly, but only answers questions from the first chapter, this may result in a high number of total answers, but probably does not reflect a meaningful learning strategy. A user with the same number of total answered questions but more unique questions presumably has learned more and therefore a higher probability for a good exam performance.

2.4.2 Maximal Chapter

As described in Section 2.1 the rules of the app require a user to work through one chapter at a time. This means that each student starts in the first chapter and must master it to a certain degree to unlock the second chapter. This process continues until the final chapter nine. When a chapter is unlocked, the corresponding badge is awarded to the student. The learning metric “Maximal Chapter” measures the highest chapter a user has unlocked. As discussed in the previous section, a learning strategy that covers higher chapters is presumably more successful regarding exam preparation than an intensive usage only in lower chapters. A student with the same number of answers but a higher number of different chapters likely prepared better for the exam than a student with the same (or a higher) number of questions that only cover the topics of the first chapters.







No.	Achievement				Badge
<i>single-level</i>					
1	Entered matriculation number				
2	Answered each question correctly once				
3	Unlocked chapter (1-9)				
<i>multi-level</i>		<i>bronze</i>	<i>silver</i>	<i>gold</i>	
4	X consecutive usage days	3	5	15	
5	X total answered questions	50	500	1000	
6	X perfect weekly challenges	1	5	10	

Figure 2: Initial Badge Design

2.4.3 Highest Streak

According to prior research, the sequence of learning days can play a decisive role in determining learning success [21]. Continuous learning is considered to be better suited for knowledge retention than concentrating the same learning time on a few days. The goal of the badge in the fourth category in Figure 2 was to promote a continuous learning strategy. Using the app for several continuous days can manifest a habit, making it easier to learn for the days to come. The extent to which this learning strategy was used is measured by the learning metric “Highest Streak”. It states the highest number of consecutive usage days of a user in each semester. The intention of this badge was that a higher streak would lead to a better exam result. This badge is the first multi-level badge in Figure 2. The three stages are 3, 5, and 15 consecutive days.

2.4.4 Total Answers

The number of total answers given provides a basic estimate of the quantitative intensity of app usage. If a student has used the app frequently, this is necessarily indicated by a high number of answered questions. In comparable studies with data from learning management systems, such a conclusion is not so simple, since, for example, documents can also be read, but the actual reading time is not necessarily reflected in the log data [8, 13]. The click on a document in an LMS is captured, but usually there is no data about how long (and focused) the document was read. However, since there is no information to be passively consumed in the app other than feedback messages after a question, there is no reason to believe that a student remains studying at a single question for an extended time after answering it. If a student uses the app more, it presumably means that he or she learns more in general. On the one hand, it is proof of the fact that the student learns directly in the app. On the other hand, more intensive app use can also be a proxy for the fact that the student generally learns more and, for example, also engages more with the other learning materials. Nevertheless, this metric is a purely quantitative measurement without a qualitative assessment of the learning strategy. For example, a student with a certain number of answers that are all false has the same value of this learning metric as a student with exclusively right answers. However, as more learning time is considered to lead to better performance, this badge was designed to motivate a higher degree of usage time [4, 27]. This badge is the second multi-level badge in Figure 2 and is awarded for the stages of 50, 500, and 1000 answers.

2.4.5 Weekly Challenges

Competitive students are considered to be more motivated to achieve a better result in the final exam [6]. Therefore, the last badge in Figure 2 should motivate to participate in the Weekly Challenge. As the Weekly Challenge is entirely voluntary and no real rewards can be earned, an intensive use of this mode is considered a good proxy for a student to be highly competitive. The influence of this badge is analyzed with the learning metric “Weekly Challenges”, measuring the number of Weekly Challenges a student participated in. This multi-level badge is awarded for the stages 1, 5, and 10 Weekly Challenges.

3. RESULTS

3.1 Evaluating Initial Badges (RQ 1)

The following analyses examine whether the badges initially implemented in the app motivate successful learning strategies. For this purpose, I examine whether the students who earned the badges achieved a better grade on average than the students who did not.

Table 3: Unique Questions

Badge	Grade Average	n
No	2.84	799
Yes	1.95	53

Table 4: Maximal Chapter

Badge	Grade Average	n
0	3.37	137
1	3.34	87
2	3.21	88
3	3.22	79
4	2.74	54
5	2.54	43
6	2.61	27
7	2.92	25
8	2.56	45
9	2.13	267

Tables 4 and 5 show the average grades of the students who received the different badges based on the learning metrics “Unique Questions” and “Maximal Chapter”. In Table 4, it can be seen that the average grade of the students who answered all 551 questions is better than that of the comparison group. A similar picture emerges for the learning metric

Table 5: Highest Streak, Total Answers, Weekly Challenges

Badge	Highest Streak		Total Answers		Weekly Challenges	
	Grade Average	n	Grade Average	n	Grade Average	n
none	3.00	539	3.38	226	3.13	432
bronze	2.43	200	2.94	284	2.55	335
silver	2.41	108	2.36	134	1.93	56
gold	2.46	5	2.21	208	2.07	29

“Maximal Chapter”. The general trend shows that the more chapters have been unlocked, the better the average grade of the students. Although the trend is not ascending in every chapter step but has a small bump at chapters 3, 6, and 7, the result suggests that students who unlock more chapters perform better in the exam.

The following analyses examine the multi-level badges in the initial badge set (see Table 5). Students are divided into four groups: Those who did not achieve even the smallest stage, and those who achieved at least bronze, silver, or gold, respectively (see Figure 2). Here, too, the average grades in the final exam are compared group by group. Again, the trend for all three badges is that the higher a student climbs on the badge levels, the better the average grade on the exam. However, the effect size differs from badge to badge. For the badge “Highest Streak” the average grade improves by 0.54 to 0.59 depending on which stage was reached. Nevertheless, it should be noted that the gold stage was only achieved by 5 students. This already indicates that the highest stage may have been set too high. The average grade difference between the students who did not achieve any badge in the category “Total Answers” and those who achieved gold status is exceptionally high. Students who have not achieved any of the three levels obtained an average grade of 3.38 while students with a gold badge had an average grade of 2.21. The badge “Weekly Challenges” also shows that students at the higher levels tend to have better average grades. However, students with a gold badge have a slightly lower average grade (2.07) than students with a silver badge (1.93). The analysis of this badge also shows (although not quite as strongly as “Highest Streak”) that the higher levels were only achieved by comparatively few students suggesting that the thresholds were set too high. In summary, however, all initially created badges seem to motivate learning strategies that lead to better grades. The first research question (see Section 1) can therefore be answered affirmatively.

3.2 Evaluating Stages of Multi-Level Badges (RQ 2)

Still, as indicated in the previous analyses, the designs of initial badges should be further investigated to answer the second research question. Since the thresholds of the multi-level badges were based on common sense but not empirically validated due to a lack of data, I analyze in the next step, whether other stages are more appropriate to provide an optimal motivational effect. For this purpose, a decision tree will be generated with the app usage data as independent variables and the exam grades as dependent variable. The idea of such a decision tree is to determine the boundaries of the learning metrics that are best suited to divide the group into different subgroups. In other words,

at what level of the metrics is the difference between the group that exceeds the level to the group that does not exceed the level the greatest. For example, if all students who answered more than 3,000 questions have a 1.0 on the exam, it would seem to make little sense to include another stage at 4,000 answered questions. Since the discussed learning metrics are at least partially related (e.g., higher maximum chapter necessarily results in higher unique questions), all learning metrics are examined simultaneously. Moreover, all badges and not only the ones originally created as multi-level are considered in the calculations. This will allow to identify whether there are reasonable thresholds for single-level badges as well, and whether they should therefore be converted into multi-level badges. The result of the corresponding decision tree is shown in Figure 3.

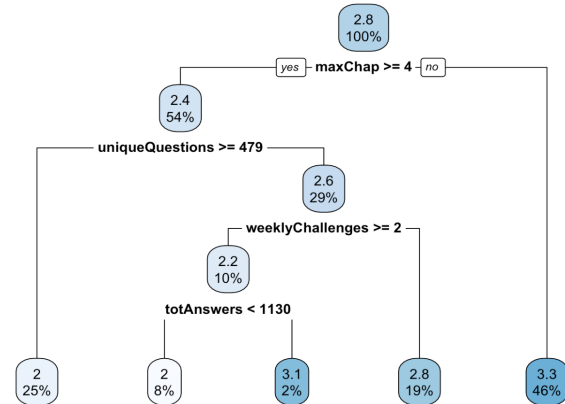


Figure 3: Decision Tree with Initial Badges

The decision tree can be interpreted as follows. “Maximum Chapter” is the most important learning metric for group classification in the data set. The average grade for all students is 2.8. For students who have only worked on chapters 1 to 3, the grade average is 3.3, and for students who have worked on at least chapter 4, the grade average is 2.4. The result can be interpreted in such a way that, for example, the badge for chapter 4 should be particularly highlighted, as it seems to mark an important threshold. In the next step, the learning metric “Unique Questions” is used for further subdivision. If students have answered at least 479 unique questions, the grade average improves to 2.0, if not, it declines to 2.6. The badge for “Unique Questions” has so far only been awarded if all questions, i.e. 551, have actually been answered. The data show that a change should be considered, and the badge should be changed from a single-level to a multi-level badge with one level at around 479. In the

next step, the group with less than 479 unique questions is subdivided according to the metric “Weekly Challenges”. If a student participated in less than 2 Weekly Challenges, the average grade worsens from 2.6 to 2.8. If the number of Weekly Challenges is at least 2, the average grade increases to 2.2. Since the smallest level of this multi-level badge was 1, the result not necessarily indicates a need for action at this point. In the last step, the group is subdivided based on the “Total Answers”. However, the subdivision goes in the opposite direction at this point. The average score improves to 2.0 if less (!) than 1,130 questions were answered. If more than 1,130 questions are answered, the average grade deteriorates to 3.1. Therefore, it does not seem to be a good strategy to simply answer as many questions as possible. However, it should be emphasized that this advice includes all the subdivisions made previously, so the limit of 1,130 answers should not be considered in isolation. In addition, it should be emphasized that “Highest Streak” was also included in the analysis but is apparently not suitable as a criterion for subdividing the groups.

3.3 Group Comparison

As discussed in Section 2.2, the data used in this study only comes from students who gave consent to sharing their usage data for research purposes. As all other usage data cannot be connected to exam results, it is omitted from the analyses in this study. This circumstance entails the risk that the corresponding results can only be interpreted considering a selection bias. Table 6 contains the result of the group comparison regarding the learning metrics of all active users that gave consent to share their usage data with the students who did not:

Table 6: Group Mean Comparison (Active Users)

Consent	Yes (n=838)	No (n=1,023)
Unique Questions	254.13	170.08
Maximal Chapter	5.42	4.04
Highest Streak	2.77	2.22
Total Answers	649.81	405.38
Weekly Challenges	1.79	0.95

The results show that there is a substantial difference between the two groups. An evaluation of every difference with a two-sided t-test showed a high significance in each comparison ($p < 0.01$). Users who have entered their matriculation number in the app and thus consented to the use of their data for research purposes are more active in the app than those who have withheld this consent. As the analyses in the preceding sections only contain the data from the subgroup who gave consent to the usage of their data, the effects of the learning metrics on exam score from the examined subsample cannot be unconditionally generalized to the whole sample due to a selection bias.

4. DISCUSSION

The objective of the present study was to develop an approach to test the design of badges in a gamified learning app with respect to the motivation of optimal learning strategies. In the first step, I examined whether the badges developed for the launch of the app motivate learning strategies that lead to better grades. This could be affirmed. In

the second step, I evaluated whether the levels of multi-level badges were optimally selected. A decision tree was developed for this purpose. The result of this analysis was that the originally selected levels could be optimized. However, as the decision tree output suggests a single threshold for each badge, a still open question is whether this threshold should be a replacement or if the future badge should only have two stages. This result could be considered in one of the next updates and in the following semester it could be checked if the newly chosen levels are optimal. This iterative process can also be applied to other learning applications. The major advantage of a decision tree here is that the results are very easy to interpret and transfer into practice. In my view, this advantage outweighs the disadvantages of this method (e.g., lower accuracy).

While the proposed process can be applied to other learning applications, the specific results for this project must be interpreted with caution. Due to the strict data protection rules at the university, I could not evaluate the data of all users, but only of those who gave us their permission to do so. This leads to a selection bias in the results. Thus, the effect of the badges cannot be confirmed for the entire cohort and therefore any adjustments to the badge architecture have to be made taking this issue into account. As previously described, one result of the study is that the levels of multi-level badges should be reconsidered. However, it could also be that certain learning strategies are not represented at all by the currently implemented badges. Therefore, in a future study, other learning strategies and the corresponding learning metrics will be developed and it will be examined whether these led to better exam results. If so, designing a corresponding badge to motivate the learning strategy would need to be considered. Until now, for example, the temporal aspect of the learning process has not been taken into account. Theoretically, the usage measured with most of the previously determined learning metrics could have taken place on one single day of the semester. More realistic are probably multiple days, but these could have been, e.g., shortly before the exam (indicating a cramming behavior) or directly at the beginning of the semester. A large body of research suggests that spaced repetition is best for optimal learning success [7]. If a student follows this strategy, he or she will have used the app on more days than a student who only started studying shortly before the exam and therefore followed a cramming strategy. This fact could be measured by a metric that states how many days in the semester a student has answered at least one question. In principle, it can be assumed that on average more learning days lead to a better performance in the exam. However, since there may also be an upper limit above which the marginal benefit decreases, a multi-level badge could be a suitable solution. Moreover, an early start could be decisive for successful course completion [23]. Therefore, the first use of the app in relation to the exam date could be measured in days. The extent of usage is measured by the metrics already discussed but this metric would measure whether a student started to use the app early in the semester. Therefore, students with an extreme cramming strategy, i.e., with the intention to start learning shortly before the exam would have a low expression of this figure. An implementation of this learning metric into the decision tree would show if there is a certain starting date that should be incentivized by a badge.

5. REFERENCES

- [1] A. Antonaci, R. Klemke, and M. Specht. The effects of gamification in online learning environments: a systematic literature review. In *Informatics*, volume 6, page 32. Multidisciplinary Digital Publishing Institute, 2019. Issue: 3.
- [2] S. Bai, K. F. Hew, and B. Huang. Does gamification improve student learning outcome? Evidence from a meta-analysis and synthesis of qualitative data in educational contexts. *Educational Research Review*, 30:100322, 2020. Publisher: Elsevier.
- [3] L. Blair. What video games can teach us about badges and pathways. *Digital Badges in Education: Trends Issues and Cases*, pages 62–70, 2016.
- [4] B. S. Bloom. Time and Learning. *American Psychologist*, 29(9):682–688, 1974. Publisher: ERIC.
- [5] A. G. Brink. The Impact of pre-and post-lecture quizzes on performance in intermediate accounting II. *Issues in Accounting Education*, 28(3):461–485, 2013.
- [6] N. E. Cagiltay, E. Ozcelik, and N. S. Ozcelik. The effect of competition on learning in games. *Computers & Education*, 87:35–41, 2015.
- [7] S. K. Carpenter, N. J. Cepeda, D. Rohrer, S. H. Kang, and H. Pashler. Using spacing to enhance diverse forms of learning: Review of recent research and implications for instruction. *Educational Psychology Review*, 24(3):369–378, 2012. Publisher: Springer.
- [8] R. Conijn, C. Snijders, A. Kleingeld, and U. Matzat. Predicting student performance from LMS data: A comparison of 17 blended courses using Moodle LMS. *IEEE Transactions on Learning Technologies*, 10(1):17–29, 2016. Publisher: IEEE.
- [9] D. Easley and A. Ghosh. Incentives, gamification, and game theory: an economic approach to badge design. *ACM Transactions on Economics and Computation (TEAC)*, 4(3):1–26, 2016.
- [10] H. Ebbinghaus. Memory: A contribution to experimental psychology. *Annals of neurosciences*, 20(4):155–156, 2013. Publisher: SAGE Publications.
- [11] S. Einig. Supporting students’ learning: The use of formative online assessments. *Accounting Education*, 22(5):425–444, 2013.
- [12] L. Facey-Shaw, M. Specht, P. Van Rosmalen, D. Brner, and J. Bartley-Bryan. Educational functions and design of badge systems: A conceptual literature review. *IEEE Transactions on Learning Technologies*, 11(4):536–544, 2017. Publisher: IEEE.
- [13] D. Gašević, S. Dawson, T. Rogers, and D. Gasevic. Learning analytics should not promote one size fits all: The effects of instructional conditions in predicting academic success. *The Internet and Higher Education*, 28:68–84, 2016. Publisher: Elsevier.
- [14] J. Hamari. Do badges increase user activity? A field experiment on the effects of gamification. *Computers in human behavior*, 71:469–478, 2017.
- [15] R. Huang, A. D. Ritzhaupt, M. Sommer, J. Zhu, A. Stephen, N. Valle, J. Hampton, and J. Li. The impact of gamification in educational settings on student learning outcomes: a meta-analysis. *Educational Technology Research and Development*, 68(4):1875–1901, 2020. Publisher: Springer.
- [16] G. James, D. Witten, T. Hastie, and R. Tibshirani. *An introduction to statistical learning*. Springer, 2021.
- [17] M. D. Kaplowitz, T. D. Hadlock, and R. Levine. A comparison of web and mail survey response rates. *Public Opinion Quarterly*, 68(1):94–101, 2004. Publisher: Oxford University Press.
- [18] J. Koivisto and J. Hamari. The rise of motivational information systems: A review of gamification research. *International Journal of Information Management*, 45:191–210, 2019. 160 citations (Crossref) [2021-04-21].
- [19] J. Langenhagen. The Use of a Gamified Learning App in Accounting Education: Exploring the Impact of COVID-19. In *International Workshop on Higher Education Learning Methodologies and Technologies Online*, pages 156–169. Springer, 2021.
- [20] C. S. Loh, Y. Sheng, and D. Ifenthaler. Serious games analytics: Theoretical framework. In *Serious games analytics*, pages 3–29. Springer, 2015.
- [21] D. Malekian, J. Bailey, and G. Kennedy. Prediction of students’ assessment readiness in online learning environments: the sequence matters. In *Proceedings of the Tenth International Conference on Learning Analytics & Knowledge*, pages 382–391, 2020.
- [22] R. McDaniel. What we can learn about digital badges from video games. In *Foundation of digital badges and micro-credentials*, pages 325–342. Springer, 2016.
- [23] S. Palmer. Modelling engineering student academic performance using academic analytics. *International Journal of Engineering Education*, 29(1):132–138, 2013. Publisher: International journal of engineering education.
- [24] M. Sailer and L. Homner. The Gamification of Learning: a Meta-analysis. *Educational Psychology Review*, 32(1):77–112, 2020. Publisher: Springer Nature BV.
- [25] L. J. Sax, S. K. Gilmartin, and A. N. Bryant. Assessing response rates and nonresponse bias in web and paper surveys. *Research in Higher Education*, 44(4):409–432, 2003. Publisher: Springer.
- [26] T.-H. Shih and X. Fan. Comparing response rates in e-mail and paper surveys: A meta-analysis. *Educational Research Review*, 4(1):26–40, 2009. Publisher: Elsevier.
- [27] J. Stallings. Allocated academic learning time revisited, or beyond time on task. *Educational Researcher*, 9(11):11–16, 1980. Publisher: Sage Publications Sage CA: Thousand Oaks, CA.

Improving problem detection in peer assessment through pseudo-labeling using semi-supervised learning

Chengyuan Liu
North Carolina State
University
cliu32@ncsu.edu

Yunkai Xiao
North Carolina State
University
yxiao28@ncsu.edu

Jialin Cui
North Carolina State
University
jcui9@ncsu.edu

Qinjin Jia
North Carolina State
University
qjia3@ncsu.edu

Ruixuan Shang
North Carolina State
University
rshang@ncsu.edu

Edward Gehringer
North Carolina State
University
efg@ncsu.edu

ABSTRACT

An online peer-assessment system typically allows students to give textual feedback to their peers, with the goal of helping the peers improve their work. The amount of help that students receive is highly dependent on the quality of the reviews. Previous studies have investigated using machine learning to detect characteristics of reviews (e.g., Do they mention a problem, make a suggestion, or tell the student where to make a change?). Machine-learning approaches to peer-assessment evaluation are heavily reliant on labeled data to learn how to identify review characteristics. However, attaining reliable labels for those characteristics is always time-consuming and labor-intensive. In this study, we propose to apply pseudo-labeling, a semi-supervised learning-based strategy, to improve the recognition of reviews that detect problems in the reviewed work. This is done by utilizing a small, reliably labeled dataset along with a large unlabeled dataset to train a text classifier. The ultimate goal of this research is to show that for peer assessment evaluation, we can utilize both unlabeled and labeled datasets to obtain a robust auto-labeling system and thereby save much effort in labeling the data.

Keywords

Peer assessment, Problem detection, Natural language processing, Semi-supervised learning, Pseudo labeling

1. INTRODUCTION

Peer assessment has long been used as a pedagogical technique in project-based courses [10, 11, 13, 14]. An online system typically allows students to provide numerical scores and give textual feedback on other teams' work. It has been shown to be remarkably effective in improving students' learning and teaming skills [10]. Peer assessment can also help instructors evaluate student work and assign

grades to it. Double et al. [3] presented a meta-analysis suggests that peer assessment improves academic performance even more than teacher assessment. However, the reliability and validity of peer assessment are completely determined by peer-review quality [15]. High-quality reviews can help authors precisely identify issues with their work and make corresponding revisions. Low-quality reviews could be unhelpful or even detrimental to students' learning.

Hence, there is a growing interest in evaluating the review quality in peer assessment research [12]. However, having the instructors or TAs evaluate or grade all peer-review comments would be extremely time-consuming. Consequently, several studies have investigated machine-based automated review evaluation with the help of natural language processing techniques as well as machine learning, Nelson et al. [12] carried out a pioneering study on identifying high-quality reviews by investigating the features in the review text and determining what type of comments are most helpful and why. Xiao et al. proposed a machine-learning NLP-based approach for finding problem statements [18] (e.g., Do they mention any problems in the work that required revisions) and suggestions (e.g., Do they provide any suggested solutions on how to revise the work) [[23]] in the peer review comments.

As with most AI tasks, the biggest challenge for applying machine learning and deep learning algorithms to peer assessment is collecting labeled data [18]. Identifying whether review comments contain problem statements and suggestions is sometimes subjective, so the same review will be labeled differently by different students. This creates a major obstacle in collecting precise and reliable labels for the text analysis. Researchers have suggested approaches to tackle this problem of unreliable and insufficient labeling. Jia et al. [8] proposed an annotation process by two graduate students and measured the inter-annotator agreement between them to improve labeling validity. Xiao et al. [18] proposed to apply transfer learning and active learning to tackle the insufficient-labeling problem by using knowledge from a related task that has already been learned. All previous approaches required intervention from either human effort or out-of-domain knowledge; there is not a single study on peer assessment evaluation that has examined how to train a robust classifier on the data alone, and in particular how to

C. Liu, J. Cui, R. Shang, Y. Xiao, Q. Jia, and E. Gehringer. Improving problem detection in peer assessment through pseudo-labeling using semi-supervised learning. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 391–397, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853127>

make good use of unlabeled dataset, which is much easier to collect. Semi-supervised learning has proven to be an effective approach to address these issues [6].

Semi-supervised learning is a learning paradigm that stands between unsupervised and supervised learning [6]. The goal of semi-supervised classification is to train a classifier that uses both a small labeled dataset and a large unlabeled dataset to outperform the traditional supervised classifier trained only on the labeled data. Basic approaches to semi-supervised learning involve a well-known technique called pseudo-labeling [9], in which method a classification model is first trained on the labeled dataset and then used to infer pseudo-labels on the unlabeled dataset. Then the unlabeled dataset with pseudo-labels is combined with the labeled dataset, so that the predicted labels are used as ground truth. This allows us to essentially scale up the labeled dataset in order to train a more robust model.

Xie et al. [19] conducted an extensive study on pseudo-labeling and presented a self-training method with “student” and “teacher” models on image classification tasks (note that “student” and “teacher” here are not referring to the user of the model), which achieved an outstanding result. The idea is almost the same as the pseudo-labeling approach. Initially a “teacher” model is trained on the labeled dataset and used to predict pseudo-labels on the unlabeled dataset, then a “student” model will be trained on the combination of the labeled and the pseudo-labeled dataset, these steps will be run iteratively. In each iteration, once the “student” model is trained, it will be used as the “teacher” model to generate predictions in the next iteration. This has proven to be a promising approach by creating more labeled data to address the data-insufficiency issue. Pseudo-labeling and self-training strategies have been widely applied in computer vision tasks [5, 19]. However, very few studies have attempted this approach in natural language processing tasks. This paper aims to apply natural language processing techniques and text-classification models to investigate the validity of applying pseudo-labeling to improve the performance of detecting characteristics in the peer review comments.

The main pedagogical contribution of this study to the peer assessment evaluation is to show how to deploy our student taggers (people who labeling the review data) more effectively and eventually build an auto-labeling system. More specifically, in our peer assessment system, the labels can only be collected from student taggers, and, if they are requested to label numerous comments, they may potentially become careless, resulting in the poor labeling quality. We could deploy them more successfully by applying the pseudo-labeling approach to build a text classifier for evaluating peer reviews with considerably less labeled data.

2. METHODOLOGY

2.1 Automated peer-review quality evaluation

Although peer review is a widely accepted approach in the educational setting, the effectiveness of peer review in promoting students’ learning can vary significantly. Most research has investigated the overall pedagogical contribution of peer review of writing. However, research on evaluating review quality is particularly lacking.

Nelson et al. [12] demonstrated that high-quality review has proven to be a great benefit in improving students’ learning. Their paper proposed an approach to determining what type of feedback is most helpful and why it is helpful to students’ writing performance. They also listed the features for identifying high-quality peer reviews. This study laid an excellent foundation for later research projects on automatically detecting characteristics of peer review comments.

The earliest study on automated peer-review quality evaluation was conducted by Cho et al. [1]. This study proposed a machine-learning algorithm to evaluate peer reviews collected from SWORD—a web-based reciprocal peer-review system. The review data was encoded for multiple characteristics such as problem detection, solution suggestion, etc., and then several traditional machine learning algorithms (Naive Bayes, SVM, and Decision Tree) were applied on the text-classification task to evaluate quality.

Subsequently, automated evaluation became increasingly fashionable in peer assessment. Xiong et al. applied supervised machine learning to automatically identify problem localization (pinpoint the location of where the problem is) [20, 22] and helpfulness [21] in peer review comments using NLP techniques. Zingle et al. [23] describe a method for automatically detecting suggestions in the review text, Xiao et al. [17] proposed to auto-detect problem statement in review comments.

Our study introduces an intriguing approach for automatically assessing review quality by detecting problem statements in the comments and applying a semi-supervised learning approach to address the problem of labeled-data insufficiency. Our goal is to help students get instant and accurate feedback on the reviews they write and enable them to improve their reviewing. This approach can also significantly reduce the workload of student taggers who label those characteristics in peer-review comments.

2.2 Semi-supervised Learning & Pseudo-labeling

Deep learning has achieved great success in the area of artificial intelligence; however, most of the state-of-the-art (SotA) models were trained using supervision, which required a large labeled dataset to attain excellent performance [6]. In most cases, labeling was a difficult and time-consuming task; even if we devote the time to do this, we would still be ignoring potential insights from the unlabeled dataset, which is far easier to collect in the real world. Semi-supervised learning has shown promise from using both labeled unlabeled data. The objective of semi-supervised learning is to improve learning behavior by combining labeled and unlabeled data, or equivalently, to achieve the same model performance with a relatively small labeled dataset.

Pseudo-labeling is one of the most effective and efficient methods in semi-supervised learning [9]. With pseudo-labeling, the initial model is trained on the labeled dataset:

$$D_L = \left\{ (x^i, y^i) \right\}_{i=1}^{N_L} \quad (1)$$

, where x^i represents each input, $y^i \subseteq \{0, 1\}$ is the corresponding labels where 0 represents “does not include prob-

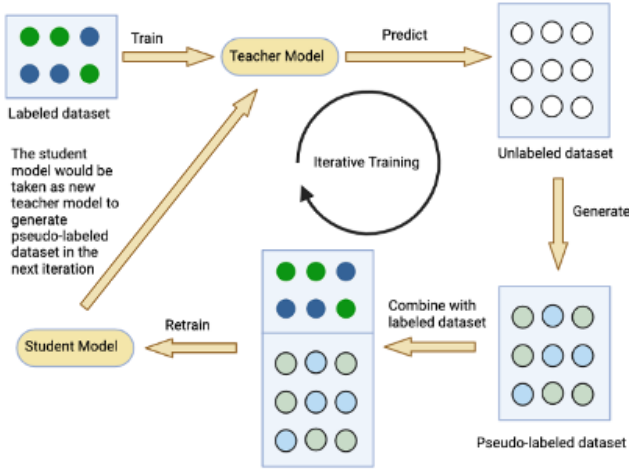


Figure 1: Self-training workflow

lem statement” and 1 represents “does include problem statement”, N_L is the size of the labeled dataset. There is also an unlabeled dataset:

$$D_U = \left\{ (x^i) \right\}_{i=1}^{N_U} \quad (2)$$

where x^i represents each input without labels, and N_U is the size of the unlabeled dataset. In most cases $N_U \gg N_L$, so we believe that the unlabeled dataset may potentially contain more valuable features than the labeled set. Next, the model trained in the initial step generates predictions \tilde{y}^i as pseudo-labels on the unlabeled set D_U ; hence we can construct a pseudo-labeled set:

$$D_U = \left\{ (x^i, \tilde{y}^i) \right\}_{i=1}^{N_U} \quad (3)$$

where \tilde{y}^i will be used as the ground-truth label y^i to compute the loss in back-propagation in the next training phase, after this, another model is trained on the combination dataset:

$$D_C = D_L + D_U \quad (4)$$

and we believe that with more training data, the model will become still better.

Pseudo-labeling is often performed through an iterative process rather than one-step label generation. Self-training [19, 5] can be interpreted as an iterative pseudo-labeling approach. Defining the model used to generate labels as “teachers” and the model trained on both pseudo-labeled and labeled dataset as “students”, the two roles will be swapped in each iteration after incorporating more pseudo-labeled data into the labeled dataset (as shown in Figure 1). In this way, we can achieve our initial goal of improving model performance with the help of the valuable unlabeled dataset without human intervention or external knowledge.

3. EXPERIMENT

3.1 Datasource

The dataset we used in this paper comes from [Redacted] [4], a web-based peer review system that allows students to provide both numerical ratings and textual feedback to

other groups’ assignments. Each student must review multiple assignments and provide appropriate peer assessments to earn credits. The scores assigned by peer reviewers help the instructor or TAs to give a final grade to the assignments. The textual feedback also helps the authors to make revisions. Students have the chance to earn extra credit by labeling the review comments they received from the peer reviewers, and these labels support the construction of text classifiers for peer-review evaluation as ground-truth labels.

Student taggers label the review comments for whether or not they contain characteristics such as: problem statements, suggestions, and explanations. In this study, we only use the problem statement label. The quality of these labels cannot be fully guaranteed, but the success of training a robust model depends heavily on the quality of the ground-truth labels. We propose a data-filtering approach to select the review comments with the most reliable labels. When students use the [Redacted] system, up to four students on a team will label the same review comments they received on their team’s work. We will not select a review comment for the dataset if any team members disagree on the label (this will be defined as “taggers agreement rule” in the following section). Initially, 48,412 review comments with the corresponding labels were pulled from [Redacted] from the Fall 2017 to Fall 2020 semesters of a masters-level object oriented design class. After the raw data was filtered by the taggers agreement rule, 3100 pieces of “high-quality” labeled data were collected. Since our goal is to investigate the effectiveness of the unlabeled set, only a small labeled subset is required to train the initial model. Because of that, we extract 1600 review comments as the training set and 1500 as the validation set. The remaining 45,312 review comments that do not follow the taggers agreement rule will have their labels stripped and used as the unlabeled dataset.

Another motivation for pseudo-labeling is to compare the effectiveness of this strategy on different sizes of labeled datasets. If the amount of labeled data required can be reduced without harming the model performance, our approach can have a great impact on peer-assessment evaluation. For this paper, we conducted multiple experiments with different sizes of labeled sets. We will report only which size brings the most improvement after applying our strategy, rather than comparing model performance between different-sized datasets, as it is an indisputable fact that more labeled data will produce a better result.

3.2 Model implementation

Comparing the performance of different deep-learning models was not a goal of this study; hence we will only select one language-classification model to train both the teacher and student models. We use the transformer-based language model known as Bidirectional Encoder Representations from Transformers (BERT), which was first introduced by Google in 2019 [2]. Transformers apply a specific self-attention mechanism, which is designed for language understanding [16]. Self-attention emphasizes which part in an input sentence is crucial to the understanding. The transformer is an encoder-decoder-based architecture consisting of a standard feed-forward layer and a special attention layer, as shown in Figure 2 [16].

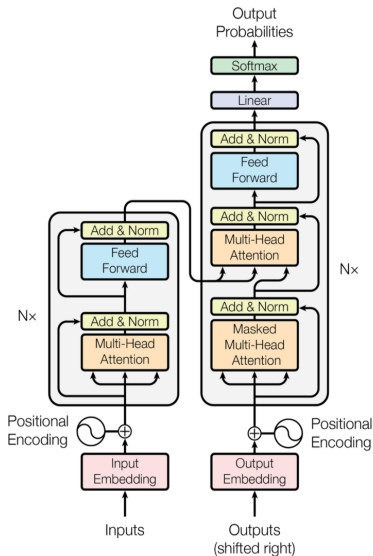


Figure 2: The Transformer - model architecture [16]

The traditional language model reads the input sentence in a single direction, either left to right, or right to left, which is enough for the task of next-word prediction. However, for a deep understanding of the sentence, the context is necessary. For a given word, considering both the previous and next token is valuable in learning the text representations, which is why the BERT model can achieve such superior performance on language-understanding tasks.

The BERT model consists of several layers of transformer blocks; the base model has 12 layers with 110 million parameters. By comparison, the large model has 24 layers with 340 million parameters [16]. Note that the BERT model only uses the encoder part of the transformer, which is responsible for reading the input text and processing it. In this study, we will only apply the BERT base model, considering the training efficiency.

The BERT model is trained in two phases, pre-training and fine-tuning. Pre-training includes two NLP tasks: Masked Language Modeling (MLM) and Next Sentence Prediction (NSP), using 3.3 billion words from Wikipedia and BooksCorpus; note that all data is unlabeled. Then the pre-trained model is used for the downstream NLP tasks in the fine-tuning phase, like text classification. In our study, we simply used the pre-trained BERT base model, then fine-tuned the model by feeding our peer-review data to carry out the text-classification task.

3.3 Pseudo-labeling setting

As mentioned in Section 2.2, we initially trained a teacher model only on the labeled dataset. We aimed to assess the improvement achieved by our strategy with three different sizes of the labeled set. Accordingly, 400, 800, and 1600 labeled reviews were randomly selected from the training set. These samples were used to train the initial teacher model and later combined with pseudo-labeling data to train the student model in the self-training loop.

After the pseudo-labels are generated on the unlabeled dataset, another attractive experiment is to investigate whether we should use the entire pseudo-labeled dataset, or just a part of it, to train the student model. There are three common approaches for selecting the pseudo-labeled subset. We define them as—

- **Full selection:** Combine the entire pseudo-labeled set with the labeled set to train the student model.
- **Random selection:** Randomly select a subset from the pseudo-labeled set and combine it with the labeled set; the labels of the remaining samples will be stripped, and those samples will be considered as the unlabeled dataset in the next iteration.
- **Top- $k\%$ selection:** Follow almost the same steps as random selection, except for the sampling method, as shown in Figure 3, the teacher model will retain the prediction score while generating the pseudo-labels, and then only the samples with the $k\%$ highest prediction scores will be selected.

For this paper, we use the Top- $k\%$ selection method to create the subset of the pseudo-labeled dataset in each iteration. This proved to be an effective way to address the confirmation-bias issue (Section 3.4), below. In our study, $k = 100\%$ (same as full selection) was chosen as a baseline, and the $k = 10, 20, 40\%$ were selected for the experiment.

As previously mentioned, pseudo-labeling is implemented as an iterative process so top- $k\%$ selection will be applied in each iteration. Once a pseudo-labeled subset has been selected, the remaining pseudo-labeled data is used as the unlabeled set and new predictions are generated from the teacher model in the next iteration. We ran this process 10 times (epochs = 10) and for consistency, the entire pseudo-labeled dataset will be fed into the model in the last iteration.

3.4 Handling confirmation bias

Machine-learning models predict incorrect labels when they are unable to learn enough patterns from the data. In pseudo-labeling, overfitting the student model to these incorrect labels predicted by the naïve teacher model is defined as confirmation bias. This leads to a significant impairment of the pseudo-labeling strategy. Initially, the teacher model could well be affected by noise, especially with very little labeled data being trained. Although this cannot be avoided fundamentally, there are still some approaches that can help reduce the effects of confirmation bias.

3.4.1 Top- $k\%$ selection

As mentioned in section 3.3, random selection would potentially perform better than full selection as it can alleviate the negative impact of bias, since only a subset of the pseudo-labeled data will be fed into the model in each iteration. In this way, relatively less bias will be introduced into the model.

Instead of randomly selecting the subset, the top- $k\%$ selection method is based on the prediction scores generated

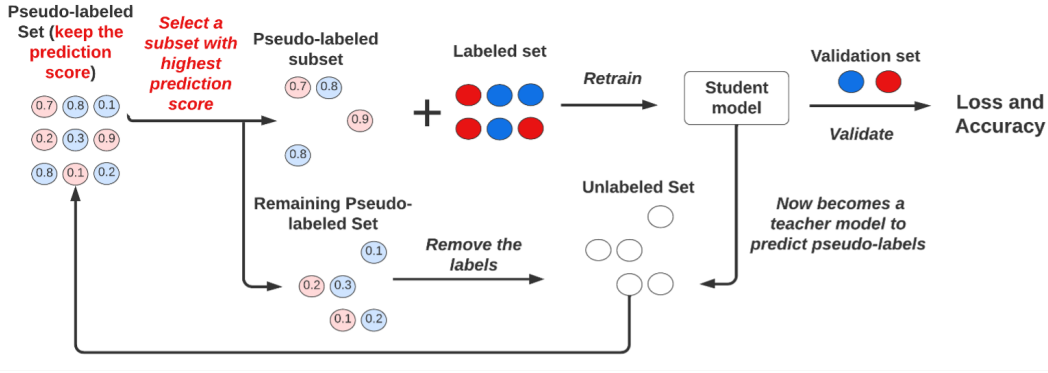


Figure 3: Top- k % selection workflow

by teacher models. Only the data points with the highest predicted probability score will be selected and included into the labeled set. The theoretical justification for this is similar to entropy regularization [7], which is another semi-supervised learning technique that encourages the classifier to infer confident predictions on the unlabeled dataset. For example, we would prefer to assign the unlabeled data a high probability of belonging to a particular class, rather than diffuse probabilities across different classes. However, this confidence-based approach must assume that the data are clustered according to class, which means that neighboring data points should have the same class, while the points in different classes should be widely separated.

3.4.2 Weighted loss

Another approach to handling confirmation bias is to re-define the cross-entropy loss as a weighted summation between the labeled and pseudo-labeled set. Initially, the naive teacher model is incapable of generating reliable pseudo-labels. If we simply add the unlabeled loss to the labeled loss, especially when the size of unlabeled dataset is much larger, the model tends to overfit on the unreliable pseudo-labeled data and consequently generate wrong predictions.

Therefore Lee et al. [9] proposed to use weight in the loss function. The overall loss function looks like this:

$$L = \frac{1}{n} \sum_{m=1}^n \sum_{i=1}^C L(y_i^m, f_i^m) + \alpha(t) \frac{1}{n'} \sum_{m=1}^{n'} \sum_{i=1}^C L(y_i^m, f_i^m) \quad (5)$$

In simple terms, the equation can be interpreted as follows: Loss per Batch = Labeled Loss + Weight \times Unlabeled Loss. In this equation, the weight (alpha value) is used to control the contribution of the unlabeled loss to the total loss. The value is initialized small and slowly increases during the model training. Since few training epochs are needed to fine-tune the BERT model for text-classification tasks, it does not seem necessary to define the alpha as a function of time. Therefore, in this study, we simply initialize the alpha value to be 0.1 and increase it by 0.1 in each epoch.

Considering the obstacle of calculating the loss after combining the labeled and pseudo-labeled sets, we designed the experiment as training on the pseudo-labeled set for each epoch and calibrating on the labeled set every three epochs. The

	Accuracy			F1 score		
	Initial	Final	Imp	Initial	Final	Imp
<i>Training with 400 labeled data</i>						
Baseline	85.3%	84.7%	-0.6%	83.4%	82.1%	-1.3%
k=10%	85.3%	91.8%	6.5%	83.4%	90.6%	7.2%
k=20%	85.3%	90.5%	5.2%	83.4%	87.7%	4.3%
k=40%	85.3%	90.2%	4.9%	83.4%	88.1%	4.7%
<i>Training with 800 labeled data</i>						
Baseline	88.9%	88.0%	-0.9%	86.2%	84.5%	-1.7%
k=10%	88.9%	92.8%	3.9%	86.2%	92.1%	5.9%
k=20%	88.9%	91.9%	3%	86.2%	90.6%	4.4%
k=40%	88.9%	91.5%	2.6%	86.2%	89.6%	3.4%
<i>Training with 1600 labeled data</i>						
Baseline	89.8%	89.3%	-0.5%	87.2%	86.2%	-1%
k=10%	89.8%	92.6%	2.8%	87.2%	90.3%	3.1%
k=20%	89.8%	92.1%	2.3%	87.2%	89.7%	2.5%
k=40%	89.8%	91.7%	1.9%	87.2%	88.8%	1.6%

Table 1: The improvement of accuracy and F1 score

alpha value is multiplied by the pseudo-labeled loss in each epoch and increases during the training iterations, while the labeled loss will remain the same.

4. RESULTS

Figure 4(a) displays the learning curve of validation accuracy and Figure 4(b) displays the F1 score, with different sizes of labeled data over 10 epochs. The performance of different values for k are compared in each plot. Table 1 demonstrates the measurement before and after applying the pseudo-labeling method. Our goal is to compare the improvement in each experiment setting to assess the effectiveness of the Top- k % selection approach and the impact of the labeled data size.

RQ1: Does the pseudo-labeling improve the model performance?

We can see from Figure 4 that in general the learning curve is continuously rising with each experiment setting, and all settings achieved a significant improvement in the last epoch, except where $k = 100\%$; we will analyze this in the following section. These results undoubtedly show that by applying the pseudo-labeling approach, we can obtain a robust classifier using a large unlabeled dataset.

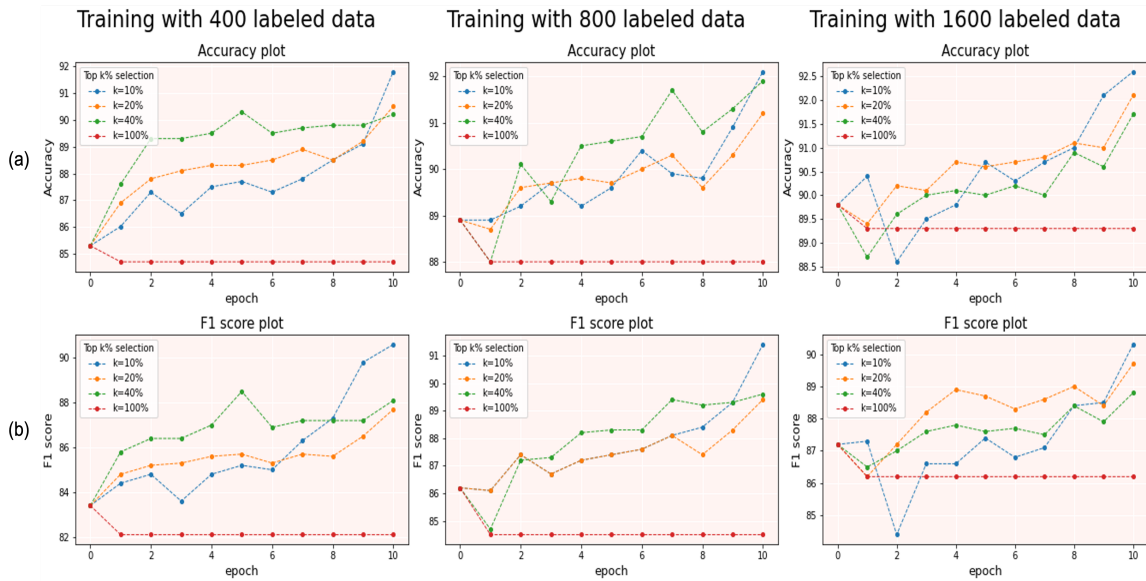


Figure 4: Validation accuracy and F1 score for different size of labeled subset

RQ2: Does the top- $k\%$ selection approach help reduce confirmation bias?

We use $k=100\%$, with the entire pseudo-labeled dataset selected as the baseline, to assess the effectiveness of the top- $k\%$ selection approach. As shown in Figure 4, the learning curve of $k=100\%$ for each setting does not change over epochs and there is a notable drop in the early epochs, indicating that the pseudo-labeling does introduce bias into the model training in the absence of top- $k\%$ selection.

Both Figure 4 and Table 1 illustrate that the improvements are slightly different with a different k value; $k = 10\%$ yields the best result regardless of the size of the labeled dataset. Table 1 shows that on an average we are able to achieve 4.4% improvement in accuracy and 5.4% improvement in F1 score with $k = 10\%$, which is higher than with the other k values. This result clarifies that by selecting the pseudo-labeled subset based on the prediction score, we can obtain high-confidence predictions as reliable labels. The result also supports the cluster assumption in our case, which means that the review comments containing the problem statement are distinguishable from the comments not containing it.

RQ3: Does the pseudo-labeling work better on a small labeled set or large labeled set?

From Table 1 we can clearly see that regardless of the k value, the overall improvement on the small labeled set is comparatively higher than on the large labeled set, which indicates that the pseudo-labeling strategy works better on the small labeled set. This also implies that the unlabeled dataset can be more valuable than labeled set in certain practical problems, and using the unlabeled set can significantly improve the learning accuracy.

5. CONCLUSIONS

This paper presents a semi-supervised learning approach based on pseudo-labeling for evaluating peer-assessment quality. We investigated the effectiveness of the pseudo-labeling

technique for different sizes of the labeled set. The results indicate that our approach can achieve an outstanding result with a small labeled dataset by augmenting it with an unlabeled set. The main contribution of this study to the peer-review process is the fact that not much labeled data is required to detect problem statements in peer-assessment comments; our student taggers do not have to label so much data. With less labeled data required, student taggers can be more careful to assign correct labels. In addition, we can find some better filtering approach to extract the smaller “high-quality” labeled data, which can greatly facilitate building our automatic labeling system.

Although we achieved a good result by using the top- $k\%$ selection approach as well as the weighted loss function to handle confirmation bias, the same success is not guaranteed on other tasks: confidence-based selection approaches are not always applicable; they will not work well without the cluster assumption.

The results of this study point the way to more efficiently analyzing review comments. Our pseudo-labeling approach can easily calculate how much labeled data for each characteristic is required for training a robust text classifier. For example, given a 91.8% classification accuracy in problem detection achieved with only 400 labeled data (a 6.5% improvement from supervised training alone), we can save a lot of labeling effort—effort that can then be devoted to identifying other salient review characteristics. Further research can explore better filtering approaches (similar to the tagger-agreement rule) for extracting small quantities of higher-quality labeled data in order to build a more reliable auto-labeling system.

6. REFERENCES

[1] K. Cho. Machine classification of peer comments in physics. *Educational Data Mining 2008 - 1st International Conference on Educational Data Mining*,

- Proceedings*, pages 192–196, 2008.
- [2] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 1(Mlm):4171–4186, 2019.
 - [3] K. S. Double, J. A. McGrane, and T. N. Hopfenbeck. The Impact of Peer Assessment on Academic Performance: A Meta-analysis of Control Group Studies. *Educational Psychology Review*, 32(2):481–509, 2020.
 - [4] E. Gehringer, L. Ehresman, S. G. GConger, and P. Wagle. Reusable Learning Objects Through Peer Review: The Expertiza Approach. *innovate Journal of On-Line education*, 3(5), 2007.
 - [5] S. Ghosh, S. Kumar, J. Verma, and A. Kumar. Self Training with Ensemble of Teacher Models. 2021.
 - [6] X. Goldberg. *Introduction to semi-supervised learning*, volume 6. 2009.
 - [7] Y. Grandvalet and Y. Bengio. entropy minimization: Semi-supervised Learning by Entropy Minimization. 2002.
 - [8] Q. Jia, J. Cui, Y. Xiao, C. Liu, P. Rashid, and E. Gehringer. ALL-IN-ONE: Multi-Task Learning BERT models for Evaluating Peer Assessments.
 - [9] D.-H. Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. *ICML 2013 Workshop: Challenges in Representation Learning*, pages 1–6, 2013.
 - [10] H. Li, Y. Xiong, C. V. Hunter, X. Guo, and R. Tywoniw. Does peer assessment promote student learning? A meta-analysis. *Assessment and Evaluation in Higher Education*, 45(2):193–211, 2020.
 - [11] K. Lundstrom and W. Baker. To give is better than to receive: The benefits of peer review to the reviewer’s own writing. *Journal of Second Language Writing*, 18(1):30–43, 2009.
 - [12] M. M. Nelson and C. D. Schunn. The nature of feedback: How different types of peer feedback affect writing performance. *Instructional Science*, 37(4):375–401, 2009.
 - [13] K. Topping. Peer Assessment between Students in Colleges and Universities Author (s): Keith Topping Source : Review of Educational Research , Autumn , 1998 , Vol . 68 , No . 3 (Autumn , 1998), Published by : American Educational Research Association Stable URL : . 68(3):249–276, 1998.
 - [14] K. J. Topping. Peer assessment. *Theory into Practice*, 48(1):20–27, 2009.
 - [15] M. van Zundert, D. Sluijsmans, and J. van Merriënboer. Effective peer assessment processes: Research findings and future directions. *Learning and Instruction*, 20(4):270–279, 2010.
 - [16] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017-Decem(Nips):5999–6009, 2017.
 - [17] Y. Xiao, G. Zingle, Q. Jia, S. Akbar, Y. Song, M. Dong, L. Qi, and E. Gehringer. Problem detection in peer assessments between subjects by effective transfer learning and active learning. (Edm):516–523, 2020.
 - [18] Y. Xiao, G. Zingle, Q. Jia, H. R. Shah, Y. Zhang, T. Li, M. Karovaliya, W. Zhao, Y. Song, J. Ji, A. Balasubramaniam, H. Patel, P. Bhalasubramanian, V. Patel, and E. F. Gehringer. Detecting Problem Statements in Peer Assessments. 2020.
 - [19] Q. Xie, M. T. Luong, E. Hovy, and Q. V. Le. Self-training with noisy student improves imagenet classification. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2020.
 - [20] W. Xiong and D. Litman. Identifying problem localization in peer-review feedback. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6095 LNCS(PART 2):429–431, 2010.
 - [21] W. Xiong and D. Litman. Automatically predicting peer-review helpfulness. *ACL-HLT 2011 - Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 2(2009):502–507, 2011.
 - [22] W. Xiong, D. Litman, and C. Schunn. Assessing reviewers’ performance based on mining problem localization in peer-review data. *Educational Data Mining 2010 - 3rd International Conference on Educational Data Mining*, pages 211–220, 2010.
 - [23] G. Zingle, B. Radhakrishnan, Y. Xiao, E. Gehringer, Z. Xiao, F. Pramudianto, G. Khurana, and A. Arnav. Detecting suggestions in peer assessments. *EDM 2019 - Proceedings of the 12th International Conference on Educational Data Mining*, (Edm):474–479, 2019.

Evaluating Gaming Detector Model Robustness Over Time

Nathan Levin¹, Ryan S. Baker¹, Nidhi Nasiar¹, Stephen Fancsali², Stephen Hutt¹

¹University of Pennsylvania

²Carnegie Learning, Inc.

rybaker@upenn.edu

ABSTRACT

Research into "gaming the system" behavior in intelligent tutoring systems (ITS) has been around for almost two decades, and detection has been developed for many ITSs. Machine learning models can detect this behavior in both real-time and in historical data. However, intelligent tutoring system designs often change over time, in terms of the design of the student interface, assessment models, and data collection log schemas. Can gaming detectors still be trusted, a decade or more after they are developed? In this research, we evaluate the robustness/degradation of gaming detectors when trained on older data logs and evaluated on current data logs. We demonstrate that some machine learning models developed using past data are still able to predict gaming behavior from student data collected 16 years later, but that there is considerable variance in how well different algorithms perform over time. We demonstrate that a classic decision tree algorithm maintained its performance while more contemporary algorithms struggled to transfer to new data, even though they exhibited better performance on unseen students in both New and Old data sets by themselves. Examining the feature importance values provides some explanation for the differences in performance between models, and offers some insight into how we might safeguard against detector rot over time.

Keywords

Intelligent Tutoring Systems, Gaming the System, Detector Rot

1. INTRODUCTION

Adaptive systems like intelligent tutoring systems (ITSs) depend on inferential models to understand and respond to individual students. Some of these systems and models have now been applied to modeling knowledge and behavior for decades [7]. For instance, Bayesian Knowledge Tracing (BKT) models have been used in ITSs for almost 30 years [11]. Even the use of more complex machine-learned models now has an extensive history; for example, gaming the system models which predict when students are attempting to find ways other than learning to advance through the system [2, 5, 26], have been in use for 18 years.

Gaming the system models are used for several purposes, including evaluating the quality of content [13, 24], research on the longitudinal impacts of disengagement [1, 34], and automated intervention [3]. Even as ITSs have become more adaptive and user interfaces have become more engaging, students have continued to find ways to disengage from these systems [38].

While BKT models are easily and frequently refit in industrial practice, models of constructs like gaming the system require

supplementary data collection beyond the standard logged data stream, in order to create new training labels. As a result, they are expensive to fit and are not refit often. Is this a dangerous practice?

An analogy can be made to code rot, a phenomenon in computer software where over time systems degrade in performance due to their reliance on aging library dependencies, hardware updates, and breakdown in the structural integrity of design patterns [18]. Models developed through machine learning and artificial intelligence (AI) may suffer a similar fate, "detector rot", where a model stops functioning as expected over time. When a piece of code simply fails to work, it is obvious, but there may be less obvious failure modes for machine learned models. Machine Learning packages change in their functionality over time and become obsolete; just because a model is still runnable does not necessarily imply that it is functioning in the same way. This problem has been noted in machine learning research in general, where many past research results can no longer be reproduced [17].

Furthermore, even if a model is functioning the same way as it did a decade ago, that does not mean that it has not experienced a form of decay. There is more to decay in a model over time than just reproducibility. Take the Cognitive Tutor [31], the system which many of the first models detecting gaming were developed for [2, 6]. The design and interface of Cognitive Tutors have gone through significant changes over the years (and the system has been re-branded MATHia). The changes over time involve both cosmetic changes and changes in pedagogical strategies and content. We elaborate further on these changes in a dedicated section below. Students, teachers, and learning contexts also change over time. In 1995, Janet Schofield reported many students in Pittsburgh skipping lunch and staying after school to use ITSs [35], a behavior not commonly reported in U.S. classrooms today. ITSs are much more prevalent in classrooms than even 10 or 15 years ago, students and teachers are more familiar with instructional technology, and students use technology at-home more often compared to an earlier focus primarily on classroom use [16]. Students today are also much more likely to be comfortable quickly locating information on the internet and may expect this same immediacy in their interaction with ITSs [36]. As such, detector rot may be as much a problem of generalizability as reproducibility -- the model might have been completely valid in 2008, and may even still function exactly the same way, but may not be valid anymore in 2021. As such, we can and should ask: will models trained on older data (in this case, 2005) maintain accuracy when tested on current data (2021)? Beyond this, has student gaming behavior changed over the last 15 years as indicated by different features becoming more/less important when detecting gaming behavior?

This question seems on its surface to be a question about algorithm effectiveness *today*, based on historical models. But it is also important to ask, do we have any reason to believe our models will work *tomorrow*? One challenge in answering this question is that the algorithms we use today are different than those used fifteen years ago. Significant advances have occurred in machine learning algorithms over the last 16 years [20]. We can have somewhat more confidence in the potential of today's algorithms to work tomorrow,

N. Levin, R. Baker, N. Nasiar, S. Fancsali, and S. Hutt. Evaluating gaming detector model robustness over time. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 398–405, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852962>

by conducting an anachronistic form of analysis -- applying current algorithms to older data and seeing how well they work on contemporary data. One class of algorithm that has seen recent success is gradient boosted descent trees. The eXtreme gradient boosting package (xgboost) [10] has met and surpassed state of the art results across a variety of machine learning applications including the detection of gaming the system [30]. Contemporary machine learning algorithms generally achieve better predictive performance than older algorithms, as the proceedings of this conference over the last few years shows. Will contemporary machine learning models also be more robust over time than older algorithms?

To answer these questions, this paper compares the functioning of gaming detectors over time, using data sets from 2005 and 2021. We will evaluate the effectiveness of a model trained on data from 2005 and tested on data from 2021. We will also compare which features are most important within models trained on the Old data to which features are most important in models trained on the New data, to see whether the behaviors that are predictive remain consistent over time. Additionally, we will then apply a variety of machine learning models including both classic algorithms and contemporary algorithms, to determine if each of these types of algorithms is robust to changes over time.

This paper will begin by discussing how the system being studied (MATHia/Cognitive Tutor) has changed over time, and how these changes may impact the manifestation of gaming behavior. In the methods section we describe the process of obtaining training labels for newer data, the feature set implementation, and the modeling process. Next we present results, comparing models' performance over time and dig deeper into the most important features of the most effective models. We discuss potential implications for practice in the use of detectors in real-world learning systems, and finally conclude with a synthesis of our findings and potential avenues for future research.

1.1 Mathia

We studied the issue of detector rot using log data generated by Carnegie Learning's MATHia (formerly known as Cognitive Tutor -- [30]) ITS at two time periods separated by approximately 16 years (2005 and 2021). Throughout their histories, MATHia/Cognitive Tutor has been the software component of a typically blended core curricula for middle school and high school mathematics. In blended, core implementations, Carnegie Learning recommends a mix of collaborative classroom work guided by its paper worktexts (60% of instructional time) as well as individual student work (40% of instructional time) in the ITS. MATHia/Cognitive Tutor presents students with complex, multi-step math problems mapped to fine-grained skills (often also referred to as knowledge components/KCs; [23]). Within each problem, the ITS provides context-sensitive help and feedback, sensitive, for example, to particular solution strategies a student might adopt (e.g., surfacing feedback that an incorrect answer reflects an inappropriate problem-solving strategy). Implementing mastery learning [33], the ITS tracks students' progress to mastery of KCs using an implementation of Bayesian Knowledge Tracing (BKT; [11]) and presents problems to students until they demonstrate mastery of all KCs associated with each topical lesson or "workspace." When a student has mastered all KCs in a workspace, they are moved on to the next workspace in an assigned sequence of workspaces, typically corresponding to a course like Algebra I or Grade 6 Math. Within MATHia, the tutor judges a KC as mastered when the student reaches a 0.95 probability estimate for having mastered that KC.

Timestamped log data track student actions (e.g., making a problem-solving attempt, requesting a hint) at each step of problems within each workspace, as well as feedback from the ITS (e.g., a correct response or that an error triggers just-in-time feedback because it reflects a common misconception, etc.). Data also track the input values provided by students, the KC to which a particular problem-step is mapped, and BKT's ongoing estimation of a student's probability of having mastered the KC.

1.2 Mathia Changes Between 2005 and 2021

In general, the Cognitive Tutor Java application of 2005 was more compartmentalized (with multiple windows displaying problem-solving elements) than the more contemporary web-based delivery of 2021. One prominent difference in the layout of the user interface concerns the extent to which the "skillometer" for visualizing student progress to skill mastery has evolved from a display that was "expanded" by default (displaying skill names and progress to mastery) to a more visually compact representation (circles that "fill" as students make skill progress) that can be expanded by the student to see their progress in more detail.

Changes have more recently been implemented in how BKT tracks student progress to skill mastery, especially related to how the student's use of hints impacts their knowledge estimates. First, hints are now delivered in MATHia with a delay between "hint levels." After the student requests an initial hint, which typically re-states the goal for the current problem-solving step, there is a delay of a few seconds before the student can request another hint, which provides detail on how to accomplish the goal. This initial delay and short delays for each additional hint are designed to encourage reflection on the help requested and discourage students from rapidly seeking the "bottom out" hint, which provides the answer [2]. In the Cognitive Tutor circa 2005 (and for many subsequent years), a student's request for a hint on a first attempt at a problem-solving step was treated as an incorrect response, resulting in a decrease in the ITS's estimate of skill mastery. The 2021 version of MATHia only treats the "bottom out" hint that presents the student with the answer as an incorrect attempt. Correct attempts after an initial hint are now credited (i.e., skill mastery estimates increase) like immediate correct attempts, and correct attempts after mid-level hints now leave the skill mastery estimate unchanged. In addition, MATHia's BKT parameter estimates for each skill (used to determine the models' responsiveness to correct and incorrect answers) are now frequently set based on data-driven estimation techniques [32, 39] as opposed to mostly being set according to expert judgment in earlier Cognitive Tutor versions.

2. METHODS

2.1 Labeling Gaming Behavior

We obtained the data set used to develop the gaming detector in [6]. This original detector was a J48 (C4.5) decision tree classifier [29] built using training labels developed using text replays. Text replays allow coders to directly label "clips" (segments of log data), presented as a sequence of actions and their context [4]. Text replays have been used in a range of projects as a fast and accurate method to label a range of types of student behavior for classifier development within various types of learning systems [4, 6, 12, 26].

For the older data set [6], we obtained data from the PSLC DataShop [22], data set "Algebra I 2005-2006 (3 schools)", including both training labels derived using text replays and partially-distilled log data. 18,737 training labels were included in that data set.

The New data set was obtained directly from the Mathia team (this data set is not currently on DataShop, due to government-agency-level contractual restrictions on data sharing). We conducted an identical text replay approach to [6], obtaining the original text replay software from the first author of that earlier work. We used a textual sequence of student activity of a duration of 20 seconds or 8 actions (whichever occurred last) from MATHia’s log data as a clip for labeling. Every clip contained the student ID, timestamp of each action (in relation to the first action in clip), the problem name and step, student’s input, relevant knowledge/skill production and system estimate, and the outcome as assessed by system (correct, a misconception (bug), wrong answer, a request for hint (initial or a deeper level). This set of clips was then coded for gaming the system behavior.

“Gaming the system” behavior was defined as the learner misusing the system’s help and feedback to get correct answers and advance in their trajectory within the ITS [6]. A clip was labeled as gaming the system when a learner asked for hints in quick and repeated successions until the system revealed the answer, or systematically input answers rapidly until they got the correct one. For example, a student entering a sequence like "1,2,3,4,5,6,7,8" in rapid succession would be labeled as engaging in gaming behavior. For further detail on the behaviors treated as gaming the system in the coding process, please see [27].

For the more recent data set, two coders (2nd & 3rd authors) initially labeled 60 text replays to establish inter-rater reliability, and attained a kappa of 0.62, comparable to the original data set [6], and over the 0.6 cut-off often treated as standard for coding ill-defined constructs such as disengaged behavior [25]. Subsequently, the first coder labeled a total of 600 clips from which 6 were removed as unclassifiable. Out of the 594 labels, 31 were coded as ‘gaming’, and the remaining 563 as ‘not gaming’. Thus, around 5% of the total clips were coded as gaming for this data sample, which is in alignment with previously observed proportions of gaming behavior in ITS [6].

2.2 Feature Engineering

The features developed for this research are based on the original research published in [6]. In order to maintain fidelity with the original work we followed the process of creating the original features as closely as possible, but in order to make sure the features were comparable across data sets, we re-distilled the features for the original data set. The features are described in table 1.

All features were engineered on the full data-set of student log data, and then aggregated for the labeled clips. Each clip consists of a series of actions so the features were aggregated together to create a single row of data labeled as either gaming behavior or not. The aggregate columns created for each feature were: Count of non-null values, Mean, Standard Deviation, Minimum, 25th percentile, 50th percentile (median), 75th percentile, Maximum, Sum. In total, 17 features were distilled at the transaction level, and each of these 17 features was aggregated in 9 ways in the final training data. The final training data had $17 \times 9 = 153$ features.

Table 1. Gaming the System Features

Feature Name	Description
assess_CORRECT	correct answer
assess_BUG	error tracked by MATHia for just-in-time context-sensitive feedback (e.g., a known misconception, a number as input that appears in the problem but is incorrect); typically indicates a common mistake that the tutor knows how to respond to
assess_ERROR	error not tracked for feedback, i.e. less common mistakes
assess_INITIAL_HINT	first-level hint provided
assess_HINT_LEVEL_CHANGE	a "deeper" level of hint provided
pknow	The probability estimate that the student knows this skill based on internal Bayesian Knowledge Tracing model of the student's mastery of this skill
pknow_direct [8]	If the current action is the student’s first attempt on this problem step, then pknow-direct is equal to pknow, but if the student has already made an attempt on this problem step, then pknow-direct is -1.
duration	How many seconds the action took
duration_sd	duration expressed in standard deviations from the mean time taken for this problem step across all problems
duration_sd_prev {3,5}	sum of duration_sd for previous 3 and 5 actions respectively
wrong_attempts	total number of times a student has gotten this problem step wrong (including within past problems)
error_perc	percentage of past problems the student has made errors on this same problem step
help_and_errors_count	number of times the student asked for help or made errors on this skill across all previous problems
num_steps	count of attempts on this step for this problem
help_attempts_last8	How many times has the student asked for help in their last 8 actions
error_count_last5	How many errors the student has made in the last 5 actions (includes both BUG and ERROR)
problem_step_count_last5	how many of the last 5 actions involved the same problem step

2.3 Modeling

As in [6], we modeled gaming detection as a binary classification problem - a clip with gaming the system was labeled as 1, and

without as 0. We conducted three overall types of comparisons. First, we trained models on the Old data and tested them on the Old data (Old to Old). Second, we trained models on the New data and tested them on the New data (New to New). For the Old to Old and New to New comparisons, we used a 4-fold student-level cross validation in which we left out 25% of students from each training set. We then tested each model on the left-out set of students, pooled the labels from each split, and calculated metrics on the pooled labels. In our third comparison, we took a model trained using all of the Old data and tested it on all of the New data (Old to New), using the entire training set since there was no risk of any students being present in both data sets, given the 16 year gap. We did not train a model on the New data and test it on the Old data, as doing so would not answer our research questions. There are cases where it may be of interest to conduct the ahistorical analysis of training on newer data and testing on legacy data -- such as cases where labels cannot be obtained for past data -- but it is not relevant to this use case, since text replays can be conducted on legacy data.

There was considerable imbalance between the classes in the labeled data. 5.5% of clips in the Old data set were labeled as gaming behavior, and 5.2% of clips in the New data set were labeled as gaming behavior. In order to account for this imbalance, we over-sampled the minority class to achieve a 50-50 balance between the classes, in the training sets only (not in the test sets). For over-sampling the minority class we used Synthetic Minority Oversampling Technique (SMOTE) [9] to synthesize new training data, without undersampling the majority class, to preserve all data. We used the area under the receiver operating characteristic curve (AUC ROC) to evaluate not just the predictive accuracy of our models but also the performance of our model at all classification thresholds. AUC ROC is thought to be better at evaluating classifiers in cases with strong imbalance [19], as is seen here. Gaming detection probabilities are frequently used in research involving detectors [28, 30] rather than using a single threshold; AUC ROC indicates how effective a model is across confidence levels.

We applied a variety of classic (available at the time of the original publication of the gaming detector [6]) and contemporary machine learning models on both the Old and New data sets. We were uncertain that the specific original algorithm used in [6], the WEKA J48 implementation of C4.5 incorporated into RapidMiner 4.6, could be replicated exactly at this point, so scikit-learn’s DecisionTreeClassifier, which implements the similar algorithm CART (Classification and Regression Trees), was selected as a close substitute. Scikit-learn’s implementations of Neural Networks, Random Forest, and XGBoost were also used. Of these, only XGBoost was completely unavailable in 2008 [10]. All code for this research is available for reference on github at <https://anonymous.4open.science/r/CogTutorGamingDetectors-627E>.

3. RESULTS

3.1 Model Performance

The results of our analyses evaluating different classification models in our three training-testing scenarios are shown in Table 2. In the table columns, we can see the different combinations of training and testing.

All of the classifiers performed well when trained on the Old data and also tested on the Old data (Old to Old). The best performance was obtained by Random Forest, achieving an AUC ROC of 0.784. XGBoost was second-best with an AUC ROC of 0.763, and Decision Tree was third-best, performing 0.048 worse than Random Forest. We evaluated the statistical significance of the difference

between Random Forest and Decision Tree (the algorithm closest to the original paper), using the method outlined in [14] to conduct a Z test to compare the area under two ROCs. In this case, Random Forest was a statistically significant improvement over Decision Tree, $Z = 2.334$, two-tailed $p < 0.05$.

When detectors were developed for the New data and tested on the New data (New to New), performance was generally higher than for Old to Old, rising above 0.85 for Random Forest, XGBoost, and Neural Network. However, for Decision Tree the improvement was negligible, rising from 0.736 to 0.738. Random Forest still obtained the best performance out of any of the models -- an AUC ROC of 0.929 for New to New, statistically significantly better than the 0.784 obtained in the Old to Old model, $Z=3.764$, two-tailed $p<0.001$. Decision Tree, the algorithm closest to the algorithm used in the original paper, was the only model which did not improve significantly in performance on the New to New data set, $Z=0.03$, two-tailed $p=0.973$.

Table 2. ROC AUC for different models

Model	Trained on Old Tested on Old	Trained on New Tested on New	Trained on Old Tested on New
Decision Tree	0.736	0.738	0.716
Random Forest	0.784	0.929	0.509
Neural Net	0.649	0.879	0.398
XGBoost	0.763	0.921	0.333

Our primary research question was whether gaming detector models would degrade over time. This would be shown if the Old models achieved poorer performance when applied to New data (Old to New), compared to the within-year Old to Old and New to New comparisons. All three newer models showed some degradation in performance, but there was substantial difference in degradation between algorithms. The Old to New performance for Decision Tree (AUC ROC = 0.716) appeared to have a small decline in performance relative to Old to Old (AUC ROC = 0.738, a 0.022 decline) but the difference was not statistically significant, $Z=0.360$, two-tailed $p = 0.719$. The Old to New performance for Decision Tree (AUC ROC = 0.716) was also not significantly lower than the New to New performance (AUC ROC = 0.736), $Z=0.241$, two-tailed $p=0.810$, though again there was some appearance of slightly poorer performance.

By contrast, the Old to New performance for Random Forest (AUC ROC = 0.509) was statistically significantly worse than the Old to Old Performance (AUC ROC = 0.929), $Z=6.948$, two-tailed $p<0.0001$. It was also significantly worse than the New to New performance for that algorithm (AUC ROC = 0.784), $Z=3.380$, two-tailed $p<0.001$. The Old to New performance for Neural Network (AUC ROC = 0.398) was significantly worse than Old to Old Performance (AUC ROC = 0.649), $Z=4.445$, two-tailed $p<0.001$. It was also significantly worse than the New to New performance for that algorithm (AUC ROC = 0.879), $Z=6.826$, two-tailed $p<0.001$. The Old to New performance for XGBoost (AUC ROC = 0.333) was the worst of all, significantly worse than the Old to Old Performance (AUC ROC = 0.763), $Z=6.498$, two-tailed $p<0.001$. It was also significantly worse than the New to New performance for that algorithm (AUC ROC = 0.921), $Z=7.926$, two-tailed $p<0.001$.

All four algorithms were able to achieve much better than chance performance in the Old to Old as well as the New to New scenarios, but the three newer algorithms struggled to make predictions about the New data when trained on the Old data. Decision Tree was the only model able to transfer from Old Data to the New without dropping substantially in performance. Decision Tree's performance was essentially equal when applied to unseen students in the same data set and unseen students in a new data set, suggesting that it may not have overfit to the features of the learning system/population it was being applied to. The other three algorithms (all of them less conservative algorithms than Decision Tree) performed significantly worse when comparing Old to Old performance and Old to New performance. The drops in performance on newer models and the relative robustness of the more classic decision tree model indicates that not all algorithms may be equally prone to detector rot.

3.2 Feature Importance

To understand how gaming the system is associated with student behavior in the logs, and whether this differs between time periods, we examined the feature importances of XGBoost, the algorithm with the worst drop in performance (also the newest) and Decision Tree, the algorithm with the least drop in performance (also the closest to the original paper). In doing so, we compared the models trained on the Old data and New data. Doing so can also provide evidence on how student behaviors have changed or remained consistent over time. The XGBoost algorithm calculates the importance of each feature as the 'gain', i.e. "the improvement in accuracy brought by each feature across all splits the feature is used in" [39]. In the figure below we can see the top 15 features ranked by gain in both the Old and New models, for XGBoost.

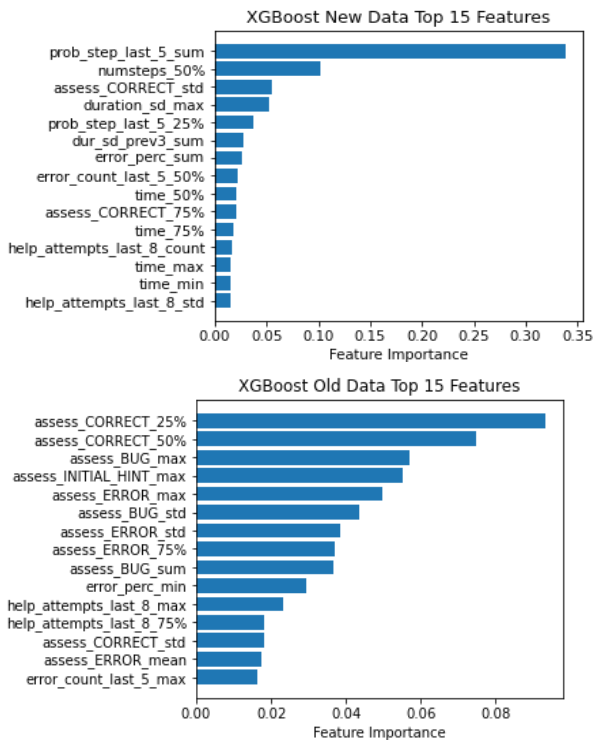


Figure 1. XGBoost Old and New Top Features

For XGBoost, in the Old Data we see that the majority of the predictive power is taken up by the features generated by the ITS's response to the student based on the answer the student has given. In particular, we see that rates of correct (assess_CORRECT)

responses and incorrect (assess_ERROR, assess_BUG) responses throughout a clip are strong predictors of gaming behavior, as well as the use of hints.

In the XGBoost model for New data, we see features that are more related to time. The sum of the trailing count of how many of the last 5 steps were on the same problem (prob_step_last_5_sum) was the most important feature. This feature indicates that a student is taking many actions on the same problem step, which could indicate that they are trying to game the system by attempting to guess the correct answer. In the other top features, we see the average time spent on the previous 5 actions in standard deviation units based on the distribution of time spent on these problems by all students (dur_sd_prev5_mean), the minimum time spent on any action within the clip (time_min), and the average time spent on each action in standard deviation units (duration_sd_mean). This is more in line with features developed in previous gaming detectors [4, 3]. Additionally, we see more of a focus on student behavior features, like error counts and wrong attempts. This represents a contrast to the XGBoost model trained on the Old Data, where most of the features were derived from the correctness of student responses and whether their errors reflect common errors (perhaps reflecting genuine errors) or rarer ones (perhaps reflecting systematic guessing). The features around bugs may be vulnerable to change over time, as the list of bugs (and the messages in response to them) has changed over the years. Features around hints could also have been impacted by changes in hint message content (which may impact learning and therefore how often they are used by non-gaming students) and by the changes to credit given to non-bottom-out hints.

By comparison, when we look at the features used by Decision Tree, we see a very different pattern. In figure 2 we see that although features such as student rates of within-clip correct, incorrect, and bugs are relevant within the Decision Tree model for the Old data (as with XGBoost), the variety of types of features being used by the Decision Tree model built on Old data is broader than for XGBoost. For instance, we see a feature representing whether or not this is the student's first time attempting a particular problem step attempt on a problem within the clip (prob_first_att_max) in the top 6. We also see the 8th most important feature was pknow-max, assessing how high the student's mastery of the best-known KC in the clip is. These features, which are not present in the XGBoost model of the Old data, are helpful in understanding the student's relationship with the problems they are facing in a given clip. Overall, comparison of feature importances indicates that the decision tree was making predictions from more disparate features than XGBoost.

When trained on the New data, the Decision Tree focused on a very small group of features that were similar to the features most important to XGBoost when trained on the New data. Again, we see the sum of the trailing count of how many of the last 5 steps were on the same problem step (prob_step_last_5_sum) as the most important feature in the New data. In the case of the Decision Tree trained on New data, this feature was the most important by a wide margin. In the next four features there are two related to evaluating the correctness of student responses (assess_CORRECT) and two that are related to the number of errors made by students in the clip (error_count_last_5_50%, error_perc_sum). These features also showed up in the XGBoost feature importance table, although at slightly different positions. The relatively stable performance of Decision Tree may be due to the stability in the meaning of the correctness assessments as opposed to the bug and error assessments (which may have shifted more in meaning between versions, with errors becoming bugs as more errors were identified).

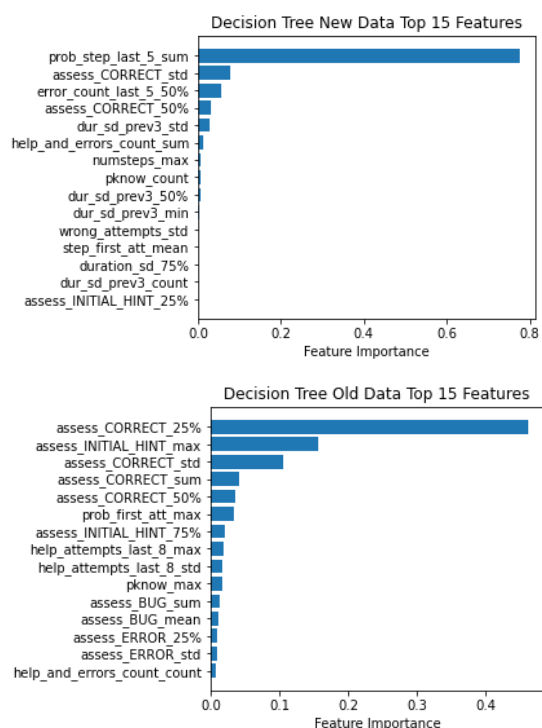


Figure 2. Decision Tree New and Old Top Features

4. DISCUSSION AND CONCLUSION

Our primary research question was the degree to which models of a complex educational phenomena such as gaming the system can be trusted over time. To investigate this question, we analyzed whether gaming detectors built on data from over 15 years ago can make reliable predictions for contemporary data. To our pleasant surprise, an older model (Decision Tree) trained on older data (from 17 years ago) still functioned well on contemporary data. However, newer, less-conservative algorithms performed much more poorly when trained on older data and tested on newer data, a phenomenon we term “detector rot”. Across all of our newer models, we observed significant rot -- significant degradation in prediction -- when training on Old data and predicting on New data.

Initial findings conducted with training and test sets from the same year initially looked positive for the newer algorithms. More specifically, Random Forest and XGBoost were able to outperform the other algorithms in both the Old to Old and New to New scenarios. This corresponds to other findings that contemporary machine learning models can offer a better fit and better cross-validated performance for gaming detection e.g. [29]. However, XGBoost was the worst-performing algorithm when trained on Old data and tested on New data. This result (plus the considerable degradation seen for the Neural Network algorithm) raises the concern that more advanced models may generally have more difficulty when applied to future samples or data drawn from different contexts.

This set of findings has important implications for detectors of complex phenomena currently in place, for the detectors being developed today, and for best practices when retraining models. We suggest the community should be cautious in using newer machine learning models -- they may initially be more accurate (even for unseen students) but may become less accurate more quickly over time than simpler models. At minimum, models developed using contemporary algorithms may need to be re-checked more often

than models using classic algorithms. However, it is not yet clear how often new data should be collected or whether old and new data should be combined (see, for instance, [20]).

In our modeling attempts most features were not important to the models. Future research might look to analyze these features and remove some of the redundancy to reduce overfitting. It is possible that this method of reducing overfitting may reduce some of the overfit to specific years, but the prominence of specific features involving student errors in the model suggests that changes in semantics between the 2005 and 2021 datasets may have been a bigger part of the explanation for the observed detector rot. It is important to acknowledge that it is not clear from our findings which changes between the Old and New data sets resulted in the detector rot observed. Across the span of 17 years, changes in the user interface, updates to the content of the ITS, and changes in student behavior may have impacted the ability of the gaming detectors to transfer. It may be worth attempting to directly identify how specific design changes impact detector performance -- for instance, by collecting text replays from right before and right after a design change. This might help understand exactly how feature importance and model functional form shifts due to this type of change, eventually helping us develop detectors resistant to these shifts and understand which design changes may reduce the effectiveness of existing detectors.

Our findings open a broad range of questions to further research on detector rot. Gaming the system is one of many classification tasks in educational data mining research and practice. Future research should investigate whether other important EDM classification problems such as drop-out/stop-out prediction and affect detection are impacted by detector rot. There is already evidence for one form of detector rot in the case of MOOC stop-out: classifiers trained on the first session of a MOOC can be less effective in later sessions [8, 36]. However, this finding may be due to differences in the populations of students who choose to take a MOOC in its first session, rather than the degradation of detectors over time -- i.e. selection bias rather than detector rot. Studying what systems and detection tasks are most prone to detector rot would be an important contribution to the practical use of detectors in real-world settings.

One of the exciting aspects of educational data mining over the last decade has been the rapid developments in the algorithms available for us to use. Newer algorithms offer the promise of better predictive performance on long-standing problems. There is a temptation to always go with the newest, most exciting algorithm available, and to focus on cross-validated performance or a held-out test set from the current data set, rather than looking at replication and generalizability (see discussion in [14]). However, our findings suggest some of our predictive models may be aging, and this may be a more serious problem for contemporary algorithms which achieve higher initial performance. Future work can help us understand which changes in learning systems and student populations result in detector rot, and how to develop adaptive and future resistant models that will support learners now and for years to come.

5. ACKNOWLEDGMENTS

We thank Carnegie Learning for providing us the newer data set for analysis, and thank the school district for agreeing to our use of the data and supporting our research. We also thank the PSLC DataShop for making earlier data available. We thank the NSF for their support of this research through grant #DUE-2000405.

6. REFERENCES

- [1] Almeda, M.V. and Baker, R.S. 2020. Predicting Student Participation in STEM Careers: The Role of Affect and Engagement during Middle School. *12*, 2 (2020), 15.
- [2] Baker, R.S., Corbett, A.T., Koedinger, K.R. and Wagner, A.Z. 2004. Off-Task Behavior in the Cognitive Tutor Classroom: When Students “Game the System.” *6*, 1 (2004), 8. DOI: <https://doi-org.proxy.library.upenn.edu/10.1145/985692.985741>
- [3] Baker, R.S.J. d., Corbett, A.T., Koedinger, K.R., Evenson, S., Roll, I., Wagner, A.Z., Naim, M., Raspat, J., Baker, D.J. and Beck, J.E. (2006), Adapting to When Students Game an Intelligent Tutoring System. *Intelligent Tutoring Systems*. M. Ikeda, K.D. Ashley, and T.-W. Chan, eds. Springer Berlin Heidelberg, 392–401. DOI:https://doi.org/10.1007/11774303_39
- [4] Baker, R. S., Corbett, A. T., & Wagner, A. Z. (2006). Human classification of low-fidelity replays of student actions. In *Proceedings of the educational data mining workshop at the 8th international conference on intelligent tutoring systems* (Vol. 2002, pp. 29-36).
- [5] Baker, R., Walonoski, J., Heffernan, N., Roll, I., Corbett, A. and Koedinger, K. 2008. Why Students Engage in “Gaming the System” Behavior in Interactive Learning Environments. *19*, 2 (2008), 185–224.
- [6] Baker, R.S. and de Carvalho, A. 2008. Labeling student behavior faster and more precisely with text replays. *Educational Data Mining*. (Jun. 2008), 38.
- [7] du Boulay, B. 2019. Escape from the Skinner Box: The case for contemporary intelligent learning environments. *British Journal of Educational Technology*. *50*, 6 (Nov. 2019), 2902–2919. DOI:<https://doi.org/10.1111/bjet.12860>.
- [8] Boyer, S. and Veeramachaneni, K. 2015. Transfer Learning for Predictive Models in Massive Open Online Courses. *Artificial Intelligence in Education*. C. Conati, N. Heffernan, A. Mitrovic, and M.F. Verdejo, eds. Springer International Publishing, 54–63. DOI: https://doi-org.proxy.library.upenn.edu/10.1007/978-3-319-19773-9_6
- [9] Chawla, N.V., Bowyer, K.W., Hall, L.O. and Kegelmeyer, W.P. 2002. SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*. *16*, (Jun. 2002), 321–357. DOI:<https://doi.org/10.1613/jair.953>.
- [10] Chen, T. and Guestrin, C. 2016. XGBoost: A Scalable Tree Boosting System. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (San Francisco California USA, Aug. 2016), 785–794. DOI: <https://doi-org.proxy.library.upenn.edu/10.1145/2939672.2939785>
- [11] Corbett, A.T. and Anderson, J.R. 1995. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modelling and User-Adapted Interaction*. *4*, 4 (1995), 253–278. DOI:<https://doi.org/10.1007/BF01099821>.
- [12] DiCerbo, K.E. and Kidwai, K. 2013. Detecting Player Goals from Game Log Files. (2013), 2.
- [13] Fancsali, S.E., Li, H., Sandbothe, M. and Ritter, S. 2021. Targeting Design-Loop Adaptivity. (2021), 8.
- [14] Fogarty, J., Baker, R. S., & Hudson, S. E. (2005, May). Case studies in the use of ROC curve analysis for sensor-based estimates in human computer interaction. In *Proceedings of Graphics Interface 2005* (pp. 129-136).
- [15] Gardner, J., Yang, Y., Baker, R. and Brooks, C. 2018. Enabling End-To-End Machine Learning Replicability: A Case Study in Educational Data Mining. *arXiv:1806.05208 [cs, stat]*. (Jul. 2018). DOI: <https://doi.org/10.48550/arXiv.1806.05208>
- [16] Gray, L. 2020. Teachers’ Use of Educational Technology in U.S. Public Schools: 2009, First Look. (2020), 70.
- [17] Gundersen, O.E. and Kjensmo, S. 2018. State of the Art: Reproducibility in Artificial Intelligence. *32*, 1 (Apr. 2018), 8. Retrieved from <https://ojs.aaai.org/index.php/AAAI/article/view/11503>
- [18] Izurieta, C. and Bieman, J.M. 2013. A multiple case study of design pattern decay, grime, and rot in evolving software systems. *Software Quality Journal*. *21*, 2 (Jun. 2013), 289–323. DOI:<https://doi.org/10.1007/s11219-012-9175-x>.
- [19] Jeni, L.A., Cohn, J.F. and De La Torre, F. 2013. Facing Imbalanced Data—Recommendations for the Use of Performance Metrics. *2013 Humaine Association Conference on Affective Computing and Intelligent Interaction* (Geneva, Switzerland, Sep. 2013), 245–251. DOI: <https://doi-org.proxy.library.upenn.edu/10.1109/ACII.2013.47>
- [20] Jordan, M.I. and Mitchell, T.M. 2015. Machine learning: Trends, perspectives, and prospects. *Science*. *349*, 6245 (Jul. 2015), 255–260. DOI:<https://doi.org/10.1126/science.aaa8415>.
- [21] Karumbaiah, S., Lan, A., Nagpal, S., Baker, R.S., Botelho, A. and Heffernan, N. 2021. Using Past Data to Warm Start Active Machine Learning: Does Context Matter? *LAK21: 11th International Learning Analytics and Knowledge Conference* (Irvine CA USA, Apr. 2021), 151–160. DOI: <https://doi-org.proxy.library.upenn.edu/10.1145/3448139.3448154>
- [22] Koedinger, K.R., Leber, B. and Stamper, J. 2010. A Data Repository for the EDM community: The PSLC DataShop. (2010), 43–56.
- [23] Koedinger, K. R., Corbett, A. T., & Perfetti, C. (2012). The Knowledge-Learning-Instruction framework: Bridging the science-practice chasm to enhance robust student learning. *Cognitive science*, *36*(5), 757-798. DOI: <https://doi.org/10.1111/j.1551-6709.2012.01245.x>
- [24] Muldner, K., Burleson, W., Van de Sande, B. and VanLehn, K. 2011. An analysis of students’ gaming behaviors in an intelligent tutoring system: predictors and impacts. *User Modeling and User-Adapted Interaction*. *21*, 1–2 (Apr. 2011), 99–135. DOI:<https://doi.org/10.1007/s11257-010-9086-0>.
- [25] Ocumpaugh, J., Baker, R.S. and Rodrigo, M.M.T. 2015. Baker Rodrigo Ocumpaugh Monitoring Protocol (BROMP) 2.0 Technical and Training Manual. (2015), 73.
- [26] Owen, V.E., Anton, G. and Baker, R. 2016. Modeling User Exploration and Boundary Testing in Digital Learning Games. *Proceedings of the 2016 Conference on User Modeling Adaptation and Personalization* (Halifax Nova Scotia Canada, Jul. 2016), 301–302. DOI: <https://doi-org.proxy.library.upenn.edu/10.1145/2930238.2930271>
- [27] Paquette, L. and Baker, R.S. 2019. Comparing machine learning to knowledge engineering for student behavior

- modeling: a case study in gaming the system. *Interactive Learning Environments*. 27, 5–6 (Aug. 2019), 585–597. DOI:<https://doi.org/10.1080/10494820.2019.1610450>.
- [28] Paquette, L., Baker, R.S. and Moskal, M. 2018. A System-General Model for the Detection of Gaming the System Behavior in CTAT and LearnSphere. *Artificial Intelligence in Education*. C. Penstein Rosé, R. Martínez-Maldonado, H.U. Hoppe, R. Luckin, M. Mavrikis, K. Porayska-Pomsta, B. McLaren, and B. du Boulay, eds. Springer International Publishing. 257–260. DOI: https://doi-org.proxy.library.upenn.edu/10.1007/978-3-319-93846-2_47
- [29] Quinlan, J.R.. (1993) C4. 5: programs for machine learning. Thousand Oaks, CA: Morgan Kaufmann.
- [30] Richey, J.E., Zhang, J., Das, R., Andres-Bray, J.M., Scruggs, R., Mogessie, M., Baker, R.S. and McLaren, B.M. 2021. Gaming and Confrustion Explain Learning Advantages for a Math Digital Learning Game. *Artificial Intelligence in Education*. I. Roll, D. McNamara, S. Sosnovsky, R. Luckin, and V. Dimitrova, eds. Springer International Publishing. 342–355. DOI: https://doi-org.proxy.library.upenn.edu/10.1007/978-3-030-78292-4_28
- [31] Ritter, S., Anderson, J.R., Koedinger, K.R. and Corbett, A. 2007. Cognitive Tutor: Applied research in mathematics education. *Psychonomic Bulletin & Review*. 14, 2 (Apr. 2007), 249–255. DOI:<https://doi.org/10.3758/BF03194060>.
- [32] Ritter, S., Harris, T. K., Nixon, T., Dickison, D., Murray, R. C., & Towle, B. (2009). Reducing the Knowledge Tracing Space. *International Working Group on Educational Data Mining*.
- [33] Ritter, S., Yudelso, M., Fancsali, S. E., & Berman, S. R. (2016, April). How mastery learning works at scale. In Proceedings of the Third (2016) ACM Conference on Learning@ Scale (pp. 71-79). DOI: <https://doi-org.proxy.library.upenn.edu/10.1145/2876034.2876039>
- [34] Sao Pedro, M.A., de Baker, R.S.J., Gobert, J.D., Montalvo, O. and Nakama, A. 2013. Leveraging machine-learned detectors of systematic inquiry behavior to estimate and predict transfer of inquiry skill. *User Modeling and User-Adapted Interaction*. 23, 1 (Mar. 2013), 1–39. DOI:<https://doi.org/10.1007/s11257-011-9101-0>.
- [35] Schofield, J. W. (1995). *Computers and classroom culture*. Cambridge University Press.
- [36] Twenge, J.M., Martin, G.N. and Spitzberg, B.H. 2019. Trends in U.S. Adolescents’ media use, 1976–2016: The rise of digital media, the decline of TV, and the (near) demise of print. *Psychology of Popular Media Culture*. 8, 4 (Oct. 2019), 329–345. DOI:<https://doi.org/10.1037/ppm0000203>.
- [37] Whitehill, J., Mohan, K., Seaton, D., Rosen, Y. and Tingley, D. 2017. Delving Deeper into MOOC Student Dropout Prediction. *arXiv:1702.06404 [cs]*. (Feb. 2017). DOI: <https://doi.org/10.48550/arXiv.1702.06404>
- [38] Xia, M., Asano, Y., Williams, J.J., Qu, H. and Ma, X. 2020. Using Information Visualization to Promote Students’ Reflection on “Gaming the System” in Online Learning. *Proceedings of the Seventh ACM Conference on Learning @ Scale* (Virtual Event USA, Aug. 2020), 37–49. DOI: <https://doi-org.proxy.library.upenn.edu/10.1145/3386527.3405924>
- [39] Yudelso, M., Fancsali, S., Ritter, S., Berman, S., Nixon, T., & Joshi, A. (2014, July). Better data beats big data. In *Educational data mining 2014*.
- [40] XGBoost developers. (2021). *Python API Reference*. XGBoost Documentation. Retrieved February 13, 2022, from <https://xgboost.readthedocs.io/>

Characterizing joint attention dynamics during collaborative problem-solving in an immersive astronomy simulation

Yiqiu Zhou

University of Illinois Urbana–Champaign
yiqiu3@illinois.edu

Jina Kang

University of Illinois Urbana–Champaign
jinakang@illinois.edu

ABSTRACT

The complex and dynamic nature of collaboration makes it challenging to find indicators of productive learning and quality collaboration. This exploratory study developed a collaboration metric to capture temporal patterns of joint attention (JA) based on log files generated as students interacted with an immersive astronomy simulation using augmented reality headsets and tablets. JA is defined as the ability to coordinate attention, which thus plays an important role in collaborative problem-solving to build the common ground for knowledge co-construction. We first developed a JA metric consisting of six distinct but closely relevant states as a measure of the collaboration process. We then conducted descriptive statistics to compare frequency and temporal pattern of JA states across three learning performance groups. Our results showed that high-learning-gain groups demonstrated visual coordination behaviors more frequently and utilized this collaboration strategy in the early stage. We then investigated sequences of these JA states, focusing on one key behavior: long and consistent shared view as a proxy for collaboration. This sequential analysis revealed two different collaboration profiles: attention follow-leader and turn takers, suggesting the existence of asymmetrical participation. Our findings indicate the potential of JA metric to predict overall collaboration quality, identify undesirable collaboration behaviors, and serve as an early warning to provide just-in-time guidance.

Keywords

Joint Attention, Shared View, Collaborative Problem-Solving, Immersive Learning Environments

1. INTRODUCTION

Collaborative problem-solving (CPS) is considered a core competency of the 21st century [14]. CPS refers to the capacity of an individual to solve a problem by sharing their knowledge, skills, and efforts with two or more people [26]. CPS provides opportunities for learners to develop the cognitive and social skills required for effective collaboration. Computer-supported collaborative learning (CSCL) environments are thus designed to facilitate this joint activity by allowing individuals to monitor collaborative progress, accommodate different perspectives, and develop a solution (e.g., [24]).

Y. Zhou and J. Kang. Characterizing joint attention dynamics during collaborative problem-solving in an immersive astronomy simulation. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 406–413, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852988>

As more advanced technologies such as augmented reality (AR) emerged, there have been challenges to understand how students use these technologies and further how technological features need to be designed to support the students' CPS process [17]. Such learning platforms provide immersive learning experiences in a classroom setting; yet it is challenging to understand their collaboration process due to the complexity (e.g., [38]). Exploring novel ways to understand collaborative learning in immersive learning environments becomes critical. In this regard, this exploratory study investigates joint attention (JA) as a proxy for collaborative behaviors in a multi-device collaborative learning platform.

JA has been studied to understand how students coordinate attention to build shared understanding in collaborative tasks. Existing studies have focused mainly on joint visual attention (JVA) using eye-tracking data and showed correlations between JVA and collaboration quality [33, 34]. However, most studies considered moments of joint visual attention (JVA) as a binary variable, which may be insufficient to capture the complex process of collaboration. We therefore developed a JA metric consisting of six distinct but closely relevant states as a measure of the collaboration process. More specifically, we investigated JA states as preliminary evidence to understand how students coordinate attention across different types of devices (AR headsets, tablets) and identify collaboration patterns that may contribute to learning gains in CPS.

2. RELEVANT WORK

2.1 CPS in immersive learning environments

CPS refers to the process when students attempt to form a shared understanding and co-construct knowledge by working on a common problem or project known as the joint problem space [32]. Advanced educational technologies like AR emerged as a medium for immersive collaborative simulations [11]. This technology brings new affordances and challenges for students to participate in CPS and for researchers to understand CPS behaviors.

Studies have suggested that immersive learning environments enhance face-to-face interaction and collaboration [39], support collaborative inquiry learning [37], and facilitate collaborative knowledge construction [21]. Pervasive AR headsets and mobile-AR systems enhance social interactions in the sense that students can collaborate through both digital devices and face-to-face interactions [4]. Immersive technologies also create a sense of immersion [9] that contributes to an authentic learning experience [39]. From this perspective, immersive learning environment fosters collaborative learning and problem-solving as it affords a dual interaction space: (1) the social interaction space through both face-to-face communication and interactions enabled by the device, and (2) the cognitive problem space by providing a simulated or 3D object that students can respond to and build knowledge on.

However, an immersive learning environment also brings challenges when it comes to understanding students' learning and collaboration processes. Given the immersive nature of this environment, it often requires expensive monitoring devices such as eye trackers, motion trackers, and video cameras to understand how students utilize these technologies [10]. How students navigate multiple representations in such an environment remains unexplored, and there is limited evidence suggesting how the navigation and exploration patterns generate learning opportunities [1]. Another challenge lies in the lack of peer awareness in immersive learning environments [7, 17], in which students are not aware of their peers' actions when they explore and solve tasks. An analysis of how they coordinate their attention to build close connections between social and cognitive problem space is fundamental to the understanding of the CPS in immersive learning.

2.2 Understanding CPS Using Log Data

Log data offers a particular advantage to examine the complex nature of human interactions in CSCL environment. While traditional quantitative methods rely on outcome variables and static variables from subjective measures, log data enables the analysis of collaboration as a dynamic process. It can capture sequences of actions and events, and thus provide an opportunity to examine collaborative learning from a process perspective. A common and significant characteristic of sequential analysis is its emphasis on the interrelations between actions over time instead of the presence or absence of the actions in isolation [20]. The application of sequential mining approaches has proven successful to identify interaction patterns differentiating low and high-achieving groups [22, 40], discover problem-solving modes in pair programming [31], identify navigation behavior patterns in the CPS process [19], and understand how regulation processes unfold over time in group work [12].

The benefits of sequential analysis are further reflected by Han et al. [15]. The authors point out the necessity to examine time-related factors when implementing collaboration analytics, as characteristics of collaboration such as coordination between group members may differ over time. Temporal aspects like sequence provide a unique perspective to understand CPS as a complex and dynamic process. However, most research efforts focused on exploring CPS supported by single platform (e.g., online learning platforms, ITS, multi-touch tablets, interactive whiteboard). There is much less research on how to discover and analyze patterns of interactions when students collaborate across multiple devices [10].

2.3 Joint Attention and Collaboration

One of the potential proxies for collaboration is JA, which is defined as "the ability to coordinate attention toward a social partner and an object of mutual interest" [3, 25]. Solving problems together requires students to share ideas and build a mutual understanding of problem-solving rules, in which students help each other think through the problem [7, 15, 28]. If JA is not achieved between partners, it is less likely for them to regulate their attention and build the necessary common ground for further discussion, and actively contribute to problem-solving. In this regard, JA is closely related to productive collaboration [27]. A prototypical example of JA involves visual synchronization, which refers to JVA—the mutual coordination of eye gaze [6]. JVA was first introduced by Scaife and Bruner [6] to study the focus of attention in infants and has been studied to understand collaboration dynamics. Previous research mainly used mobile eye-trackers to measure JVA and found correlation between JVA and dyad productivity [18], collaboration quality [33], and other outcome measures like learning gains and task performance [34]. Although these findings demonstrated the

potential of JVA to serve as a proxy for quality collaboration, high-level JVA moments may hide unbalanced participation known as the free-rider effect, suggesting the partner dominance in terms of gaze initiation [34]. Schneider and Pea [33] categorized this dyad as leader and follower, inspired by four collaboration profiles that students assume [35]: turn takers, driver-navigator, driver-passenger, and independent. This asymmetrical collaborative pattern was found negatively correlated with learning gains, as students who less frequently initiate and respond to joint objects benefit less from JVA moments. This finding points out limitations of JVA as it may hide undesirable collaboration mode and thus insufficient to measure collaboration quality.

Another limitation of existing literature is the perception of JVA as a binary and momentary event that students are either in or not in this state [18, 33, 34]. Relying on this binary classification, JVA may not accurately or sufficiently represent the process of collaboration considering its complex and dynamic nature. Siposova and Carpenter [36] argued that the jointness of attention comes in degrees rather than as arbitrary, discrete, and uniform events. They developed a systematic framework containing four levels of social attention (monitoring, common, mutual, and shared). According to their framework, attention levels are nested hierarchically and exist on a scale of jointness. It is important to distinguish between these levels as they may support different interactions and communications. To achieve a more comprehensive understanding of collaboration, it is necessary to investigate JA as a process consisting of interrelated states rather than a binary phenomenon.

2.4 Research Aim

To fill in these gaps, we conducted an exploratory study to investigate JA in a CPS process based on fine-grained log data in a multi-device immersive environment. We hope to go beyond the previous binary classification of JVA and provide insights into how students coordinate attention in the CPS process. Therefore, we designed a JA metric that consists of six different states, inspired by the spectrum of jointness framework [36]. We then explored the sequences of these states to characterize the process of attention coordination in CPS. Using a sequential analysis approach, we investigated the relationship between groups' JA states and their learning gains. Our goal is to understand how CPS unfolds over time using a sequential analysis method. By extracting key collaboration patterns that potentially lead to quality collaboration and better learning performances, we hope to characterize dynamics of JA in the context of CPS. The research questions guiding our analysis are: (1) What are temporal patterns of joint attention that are indicators of collaborative problem-solving? and (2) How do these joint attention patterns vary across groups with different learning gains?

3. METHODS

3.1 CEASAR

Connections of Earth and Sky with Augmented Reality (CEASAR) employed a digital planetarium simulation software designed to investigate collaborative learning in immersive augmented reality. CEASAR allows the exploration of the night sky through three scenes: Horizon (default), Earth, and Star. It simulates a first-person view of the night sky from a specific location and time. The Earth scene allows users to observe the entire Earth from space. Users can drop a pin on any location of the Earth's sphere to change their location or obtain its coordinate. The Star scene provides access to the complete celestial sphere and cataloged western constellations. Users can shift between these three views, manipulate the location and simulation time, or change their direction of view to explore the sky. Since this platform was designed to support

collaboration, annotations (e.g., mark a constellation) made in one device will be simultaneously visible to all users in the same group.

3.2 Participants and Tasks

This study involved 77 undergraduate students enrolled in an introductory astronomy course from a mid-western university in the United States. Students participated in three weekly one-hour sessions. The first two sessions helped students familiarize themselves with the simulation platform using gesture-controlled AR headsets (Microsoft HoloLens 2) and touch-based tablets. In the third session, 25 self-assigned groups of three to four solved a CPS task called “Lost at Sea”. Each group was provided with one AR device and two tablets. Students were expected to leverage these digital devices determine the location of a space capsule that has crashed somewhere on Earth. To complete the task, groups need to identify the hemisphere of their location, find the correct cardinal directions by identifying key stars or constellations as reference points, and estimate their latitude and longitude. Aside from the group task, students were required to complete individual pre- and post-paper-based assessments to measure their conceptual knowledge relevant to the task’s topic. Each assessment took about five minutes.

3.3 Data Source

This study explores the students’ collaboration patterns using data collected from video recordings of group work, screen recordings from the devices (Figure 1), and interactions with the simulation in the form of logs obtained from both AR and tablets. The interaction logs were recorded as rows of events, where event = {Username, Groupname, Device, Activity, Event, UTC time, Heading vectors, Simulation time, Crashsite, Location, Scene, Selected object, Selected star}. A new event was generated each time students moved their devices to change the direction of view, selected a star or constellation, chose a different scene, or manipulated the simulation time within the platform. In this study, we only focused on log features relevant to the identification of JA. The pre-/post-assessments contain one open-ended question to measure the students’ understanding of latitude and longitude calculation, which was scored by researchers from 0 to 2 based on the completeness and accuracy of students’ responses.

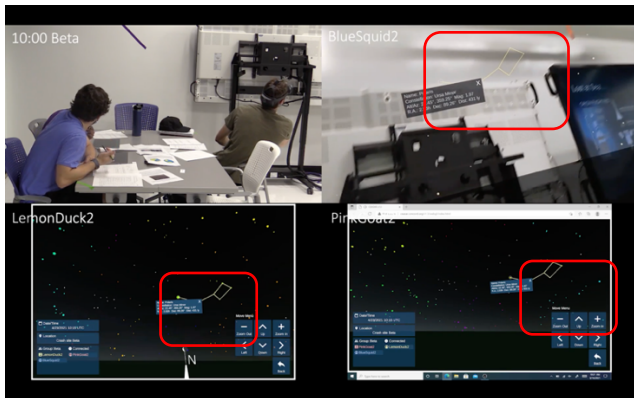


Figure 1. Screen overlap in Horizon scene across three devices—MS HoloLens2 (top right) and tablets (bottom right and left).

3.4 Data Processing

The following describes a multiple-step process of extracting JA states from raw log data.

Step 1: Individual Inactivity Extraction This step filtered active episodes and prepared for further analysis at a second granularity. We

defined 20 seconds as the threshold to distinguish active and inactive episodes. This time frame was chosen based on classroom recordings and previous study [22] showing that elapsed time beyond this threshold should be differentiated, beyond which the set of actions were perceived less relevant and supposed to belong to a different action episode. If students did not trigger any event within this 20-second time gap, subsequent seconds (i.e., from the 21st second) were labeled as inactivity until the next event. It is worth noting that during inactivity students may participate in off-task behaviors like idling or engage in task-relevant activities without using the devices, like paper sheet filling.

Step 2: Device Pair We then labeled the scene for each second, yielding a time series containing four types of scene values (Horizon, Earth, Star, and inactivity). Next, we combined the individual scene values to code the JA state of each device pair as inactivity, no overlapping, or scene overlapping. *Inactivity* means that neither device triggered any event within the 20-seconds time gap. *No overlapping* contains two possible situations: (1) two students explore the simulation in two different scenes (2) one student explores the simulation while the other is inactive. Both situations suggest a lack of JA as students engage in different activities or scenes. *Scene overlapping* represents that both students are in the same scene and observe the simulation from the same perspective. This initial coding created a state sequence for each dyad within a group, resulting in a total of three device pairs for each group (i.e., tablet1-tablet2, tablet1-HL2, and tablet2-HL2). We observed a tendency for groups to only use two devices most of the time. Considering all the dyad sequences may lead to misleading results as one device may not be used consistently and may look like a student was not engaging in the task and JA. Thus, we picked one dyad sequence that represents the whole group based on the level of participation (i.e., a dyad with the least number of inactivity and no overlapping states).

Step 3: Pair Scene Overlapping Coding This step extracted higher-level JA behavior (i.e., scene overlapping) in the horizon scene using the shared view (SV) metric (see details in [10]). SV metric tracks whether two devices’ screens overlapped, indicating students were looking at the same area of sky or celestial objects. This continuous value ranges from 0 to 1, representing the screen overlap ratio, where 1 indicates a perfect shared screen, and 0 means no overlap. By watching screen capture recordings (see Figure 1), we found that a SV value larger than 0.35 allows students to look at the same area and was thus used as the threshold to filter shared view state in Horizon scene. We also incorporated more contextual information to this state by characterizing it as ‘short’ and ‘long’ based on state duration. We chose 15 seconds as the delimiter based on our observations in terms of whether students were having a quick or in-depth longer discussion with their partners.

Step 4: Consistent State Extraction We then extracted consistent JA episodes longer than 5 seconds. Previous research pointed out that students need around 2 seconds to focus their attention on the object mentioned by their peers [30]. Considering our simulation platform requires students to manually move their screens and find the reference points or shift the scene, we set the threshold as 5 seconds. A state lasting 5 seconds or less was not sufficient to be counted as a consistent state as it may be generated by accident.

Finally, six mutually exclusive states (shown in Table 1) were identified. These states were ordered according to levels of participation and attention coordination; that is, three hierarchical levels. Each level may require varying amounts of effort to achieve attention coordination and visual synchronization. At the top level, we utilized a previously developed SV metric [10] to capture consistent *screen overlapping* behavior in Horizon. In the subsequent level,

we focused on *scene overlapping* behavior in the Earth or Star scenes. The lowest level included *no overlapping* and *inactivity* state, which could be perceived as lack of JA. JA states in Horizon was differentiated from the other two scenes and perceived as the higher level for the following reason. The current calculation method of SV metric only applies to Horizon scene, which captures the moments when students looked at the same region of the simulated sky. This is typically achieved with the help of a reference star or constellation (see the marked constellation in Figure 1). Students need to move their screens or AR headsets to find the reference object first before achieving the visual synchronization in Horizon. For the other two scenes (Star and Earth), however, students only need to click the button to select the same scene. Therefore, the extra efforts required in Horizon may indicate more verbal communication to coordinate the screens to achieve synchronization. Considering the fact that JA states differ in the level of attention and coordination, overlapping in Horizon scene is more likely to represent a high-level, intentional coordination behavior to build a shared problem space.

Table 1. JA states description

JA State		Scenes	Description
Inactivity (INACT)		N/A	Both students do not trigger any event within the 20 seconds
No scene overlapping (NO)		All	Students explore in different scenes OR one is inactive
Scene overlapping in Earth or Star (SO Earth/Star)		Earth and Star	Both students stay in Earth or Star scene
Scene overlapping in Horizon	No shared view in Horizon (SO_HZ_NO)	Horizon	Both students stay in the Horizon scene but no screen overlap
	Short-shared view (SO_HZ_SRT)	Horizon	a quick screen overlap (<15 seconds)
	Long-shared view (SO_HZ_LNG)	Horizon	a long and consistent screen overlap behavior (>15 seconds)

3.5 Group Exclusion and Classification

We used pre- and post-assessment scores as an outcome measure of learning performance. Students wrote a short response explaining the multiple steps to complete the location calculation given the visible stars and constellations, which was graded as 0, 1, or 2 based on a rubric developed during a pilot study. We computed individual normalized gains (i.e., $\text{post} - \text{pre} / \text{post-max} - \text{pre}$) to obtain each group's average normalized learning gains. Among 25 groups, the mean of normalized learning gains was 0.283 (SD = 0.290), and the median was 0.313. Six groups earned no or negative learning gains. For the rest, we conducted a median split and ended up with three performance groups: no-learning-gain ($n = 6$), low-learning-gain ($n = 9$), and high-learning-gain ($n = 10$). One group in no-learning-gain began with a full score in the pre-assessment and was removed given the ceiling effect. Four groups were removed as the students frequently shared only one device, making the logs unable to capture their collaborative behaviors. As such, a total of 20 groups were included for the further analyses: no-learning-gain ($n = 4$), low-learning-gain ($n = 9$), and high-learning gain ($n = 7$).

3.6 Analysis

Our analysis consists of two parts. First, we conducted descriptive analysis to compare aggregated values of JA states across the three learning performance groups. Examining the distribution of screen overlapping states across groups with different learning gains yields a preliminary understanding of the association between screen overlapping behaviors (i.e., JA states) and learning performance. Then we applied sequential analysis to search for patterns that characterize JA from a process perspective. Specifically, we looked at the transition probabilities between the six JA states to uncover more interesting patterns of collaboration dynamics. Two transition metrics were utilized to explore the state sequences: the Markov-chain model (MCM) and the L* metric.

MCM is a transition metric that calculates the conditional probability of one state following another based on the assumption that the occurrence probability of one state depends on the previous state. We used the TraMineR and seqHMM packages in R to build Markov models for our sequence data [13, 16]. Two important parameters in MCM are (1) the transition probabilities between the states and (2) the initial probabilities for each state. Transitions with higher probability within the sequence can be interpreted as common collaboration patterns to characterize the groups' JA dynamics. However, one limitation of MCM is the failure to take base rates into account (i.e., the initial probability of each state in the sequence). This may impact how we interpret transition probabilities and understand the relationship between states in the sequence.

We therefore applied L* metric as a complementary method. L* was chosen as the best metric for sequences without consecutive repetitive states according to the discussion in [5]. L* compares the actual occurrence probability with a calculated base rate, which is the transition probability assuming the states in the sequence are randomly ordered [23]. The use of base rates in the calculations of the L* metric makes it well suited for between-group comparison, while MCM is better suited for within-group comparison (i.e., comparison between two transitions of the same group). L* illustrates the degree to which transition between two states is more likely than in a randomly ordered sequence of states, given the base rates of each. The range of L* is $(-\infty, 1]$ where the negative value represents the specific transition is less likely to occur compared to the chance level, and 0 means this transition occurs as often as expected in a randomly ordered sequence.

4. RESULTS

4.1 Descriptive Analysis

Table 2 presents the distribution of each state across three learning levels. It suggests no-learning gain group stayed either inactivity or no overlapping states most of the time. Comparably, high-learning-gain groups had more scene overlapping states. This difference suggests that high-learning-gain groups are more likely to demonstrate JA behaviors such as screen coordination to maintain mutual attention and construct a shared problem space for in-depth discussion. We also examined the temporal aspect of data. One interesting finding was the early adoption of collaboration strategies of high-learning-gain group. We compared the long-shared view state during the first 30 states, which roughly corresponds to the first twenty minutes. While five out of seven groups in the high-learning-gain groups demonstrate long-shared view in the early stage, only one in the no-learning-gain groups and three in the low-learning-gain group demonstrate long-shared view. These results suggest that high-learning-gain groups not only demonstrate more high-level JA behavior such as screen coordination but also tend to demonstrate this behavior in the early stage of collaborative problem-solving.

Table 2. JA state descriptions in each learning gain group

State	No Gain (n=4)	Low Gain (n=9)	High Gain (n=7)
INACT	^a 19.00 ^b (29.8%)	14.78 (25.2%)	14.57 (23.9%)
NO	30.50 (47.7%)	28.00 (47.2%)	27.71 (45.2%)
SO_HZ_NO	9.25 (14.5%)	9.67 (16.8%)	12.14 (19.7%)
SO_Earth/Star	2.75 (4.2%)	3.11 (4.7%)	2.14 (3.3%)
SO_HZ_SRT	1.00 (1.6%)	2.11 (3.5%)	2.00 (3.1%)
SO_HZ_LNG	1.50 (2.3%)	1.44 (2.6%)	3.00 (4.9%)

Note. ^aAverage number of states; ^bAverage proportion of each state within each group sequence.

4.2 Sequential Analysis

We further looked at transition probabilities between states to identify collaboration patterns able to differentiate learning groups (see Figure 2). We particularly focused on the transitions relevant to the long-shared view state (SO_HZ_LNG), which is a key collaborative behavior. An interesting difference was found in the transition probabilities between the long-shared view and the other two states: no overlapping (NO) and scene overlapping in Horizon (SO_HZ_NO). These two transition sequences represent two different JA patterns, indicating to what extent groups engage in collaborative participation (e.g., symmetrical to asymmetrical participation).

SO_HZ_LNG → NO indicates that only one student remained active and interacted with the platform after the end of a higher-level JA state (i.e., long-shared view). Meanwhile, another student no longer triggered any event. By looking at the session video recordings, we found this transition typically occurred when one student initiated the screen coordination and dominated the problem-solving processes, while another student was less engaged. **SO_HZ_LNG → SO_HZ_NO** suggests a more positive collaborative behavior where pairs remained in the same scene and actively interacted with the simulation platform after leaving the screen overlapping state. Although these pairs no longer looked at the same area of simulated sky, they both continued individual exploration in a shared problem space (i.e., the same scene). We observed that this transition typically occurred when students ended discussion around the reference stars and went back to individual exploration in the same scene.

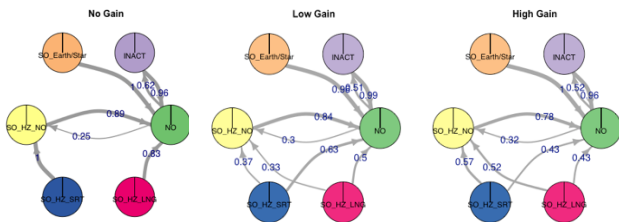


Figure 2. Markov-chain model of JA states

As shown in Figure 2, MCM presented transition probabilities within the same learning gain group. Results revealed that low-learning-gain groups showed a much higher probability for transition **SO_HZ_LNG → NO** (0.83) among all other potential transitions. Although this transition probability became smaller for low-learning-gain groups (0.5), it still remained larger compared to **SO_HZ_LNG → SO_HZ_NO** (0.33). Comparably, high-learning-gain groups showed an opposite trend. They had a higher

probability for **SO_HZ_LNG → SO_HZ_NO** (0.52) compared to **SO_HZ_LNG → NO** (0.43). This means the long-shared view state is more likely to be followed by no shared view in Horizon for high-learning-gain groups. Recall that **SO_HZ_LNG → SO_HZ_NO** suggests both students continued individual exploration after the long-shared view state, creating opportunities for information exchange and screen coordination later in the session.

Additionally, we applied the L* metric [23] to account for differences in base rates, thus allowing for between-group comparisons. When interpreting L* values, a larger absolute value indicates a stronger dependence between two consecutive states, while the value's sign (positive or negative) indicates the direction of dependence. As shown in Table 3, for **SO_HZ_LNG → SO_HZ_NO**, the high-learning-gain groups showed the highest L*, while the no-learning-gain groups showed the lowest L*. This positive value means given the previous state is SO_HZ_LNG, students in this group are more likely than chance to enter the SO_HZ_NO state. Interestingly, **SO_HZ_LNG → NO** showed the opposite trend, and the only negative value was detected in high-learning-gain groups. This means that, given SO_HZ_LNG as the previous state, the current state is less likely than chance to be NO.

In summary, L* metric can detect transitions occurring more or less frequently than random chance, providing insights about when students intentionally engage those transitions. The overall results suggest that when high-learning-gain groups exit the long-shared view state, they are more likely than chance to follow this state by entering the scene overlapping in Horizon. Similarly, when this group exit the long-shared view state, they are less likely than chance to follow this state with no shared attention at all. In contrast, this transition is around chance level (0.06) for low-gain groups or more likely than chance (0.85) for no-gain groups.

Table 3. L* Transition probabilities between long-shared view and the other states

Transition	Group	L*
SO_HZ_LNG → NO	No Gain	0.85
	Low Gain	0.06
	High Gain	-0.15
SO_HZ_LNG → SO_HZ_NO	No Gain	-0.11
	Low Gain	0.23
	High Gain	0.35

5. DISCUSSION

Our exploratory analysis identified six JA states to investigate the dynamics of JA, which provide insights into how groups coordinate their attention and solve the simulation task during a CPS process. The results revealed that groups with higher learning gains demonstrated a higher frequency of long and consistent shared view in the early stage. These preliminary findings support the previous studies that showed joint visual attention is associated with quality collaboration and contributes to learning gains (e.g., [18, 33]).

The examination of the sequence of states allowed us to identify different collaboration profiles. We observed an undesirable behavioral pattern that after a group exited a long-shared view state only one student remained active and interacted with the simulation, while another student no longer triggered any event. We perceived such different tendencies as a visual attention leader (the former) and a visual attention follower (the latter) [34, 35]. These two profiles (i.e., leader and follower), captured by the JA state sequence, illustrate the imbalanced responsibility to initiate discussion and level of engagement within the simulation. On the contrary, another sequence transitioned to individual exploration, which was more likely

to occur in the high-learning-gain groups, suggested more balanced engagement and equal responsibility of exploring the simulation environment. This transition aligns with the profile of turn takers [35], which is a more effective collaboration behavior as both participants actively engage in and maintain a joint focus on the task to solve the problem. Overall, high-learning-gain groups were most likely to demonstrate balanced, mutual collaboration while no-learning-gain groups demonstrate imbalanced participation. This detection of asymmetrical participation suggests interesting lines of follow-up inquiry on transitions of other JA states to gain a better understanding of various collaboration profiles in CPS.

The novelty and contributions of this paper lie in the following two aspects. First, we presented an exploratory study of utilizing logs to capture JA to uncover how students collaboratively solve a group task in an immersive learning environment. Understanding how students interact with immersive learning environments can be challenging due to its open-endedness, leading to unstructured interactions. This unconstrained nature of interactions makes it difficult to understand how students navigate the environment and collaborate, leading to limited evidence suggesting what collaboration patterns are associated with learning opportunities [1]. Our study contributed by developing JA metric to investigate how students coordinate their attention across devices. This method allows us to look for dynamic and fine-grained patterns of JA that characterize successful CPS and productive collaboration.

Second, we investigated JA from a process perspective, which was typically studied as aggregative values of a binary event (i.e., whether students have or not have JA) [18, 33, 34]. Our approach revealed that temporal characteristic also matters as high-learning-gain groups demonstrate visual coordination in the early stage of task session. This finding suggests that early visual coordination behavior patterns have the potential to inform the following collaboration quality. Lack of such behaviors in the beginning stage can serve as a signal for early interventions to prevent persistent undesirable or unproductive collaboration. Studies are needed for further explorations of the relationship between early collaboration patterns and following collaboration quality. Sequential analysis allows us to detect asymmetrical participation. This finding supports previous studies [34] that although high-level JVA is correlated with quality collaboration, it may also hide a free-rider effect and thus requires a finer-grained examination on this feature.

6. DESIGN IMPLICATIONS

One implication is to design learning environments in a way that facilitates the process of obtaining JA, considering the potential of JA to enhance productive collaboration. For example, we can support peer awareness by adding visual pointers like an arrow to pinpoint the direction their peers are looking at, or a coordination shortcut allowing students to synchronize the screens or scenes in the simulation quickly. Such design can facilitate coordination and visual synchronization and consequently yield quality collaboration. This is especially the case when students do not have sufficient domain knowledge to communicate the correct direction to move their screens for a shared problem space for further discussion. Given the nature of the learning environment with immersive technologies, gaze visualizations are more likely to be utilized in linguistically complex environments where it is difficult to describe reference objects or directions to look at [8]. Moreover, visual attention awareness provides evidence that other group members are engaged and indeed getting the information communicated [29]. Such awareness contributes to an improved feeling of presence [2] and encourages learners to maintain JA.

7. LIMITATIONS AND FUTURE RESEARCH

This work has several limitations that we plan to address in future studies. First, we have a small sample size containing 20 groups. Although our analysis shows interesting patterns across groups, the comparison test does not have enough statistical power to identify significant difference. Second, the L^* metric has a typical bias that inflates transition probability as our state sequence does not contain self-transition loops, which impacts the estimation of the base rate. To the best of our knowledge, transition metric for original sequences without self-transition loops is still an open issue. Current methods require the original sequences to contain self-transition loops to calculate base rates before loop removal. We also computed different transition metrics and found that MCM and L^* are the most meaningful metrics for our dataset. Our work is exploratory in nature and still in its early stage. Future research will include more participants and combine multiple data sources like video recordings and qualitative codes to better understand how collaboration unfolds.

8. CONCLUSION

This exploratory study focused on JA, a cornerstone of productive collaboration, to better understand how students regulate and coordinate their attention during CPS in an immersive learning environment. We identified different JA states and key collaboration patterns associated with learning. Specifically, we were interested in long and consistent screen overlapping across devices (i.e., long-shared view state). To advance the understanding of CPS, we applied the following approaches: (1) descriptive analysis (2) sequential analysis on JA state transition utilizing the Markov chain model and L^* metric. This preliminary exploration provides evidence that long-shared view state, representing the highest level of JA state, is closely related to students' positive collaborative learning experiences. More specifically, high-learning-gain groups demonstrate a higher frequency of long and consistent shared view in the early stage. A closer examination of the JA state sequence revealed two different collaboration profiles: attention follower and turn takers. Overall, our findings unravel the complex process of attention dynamics and yield a better understanding of attention coordination during CPS in an immersive learning environment. This understanding consequently informs the design of computer-supported collaborative learning tools and environments to enhance learning.

9. ACKNOWLEDGMENTS

First, the authors would like to formally recognize other members of our research team, Robb Lindgren, Nathan Kimball, Emma Mercier, James Planey, Taehyun Kim, and Robin Jephthah Rajarathinam. Through their design of the simulation and research insights, their effort has enriched this work significantly. The authors are also grateful to Luc Paquette for many helpful discussions and suggestions on research idea development and analysis methods. Finally, this project was sponsored by the National Science Foundation Grant no:1822796.

10. REFERENCES

- [1] Akçayır, M. and Akçayır, G. 2017. Advantages and challenges associated with augmented reality for education: A systematic review of the literature. *Educational Research Review*. 20, (Feb. 2017), 1–11. DOI:<https://doi.org/10.1016/j.edurev.2016.11.002>.

- [2] Akkil, D., Thankachan, B. and Isokoski, P. 2018. I see what you see: gaze awareness in mobile video collaboration. *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications* (Warsaw Poland, Jun. 2018), 1–9.
- [3] Bakeman, R. and Adamson, L.B. 1984. Coordinating Attention to People and Objects in Mother-Infant and Peer-Infant Interaction. *Child Development*. 55, 4 (Aug. 1984), 1278. DOI:<https://doi.org/10.2307/1129997>.
- [4] Birchfield, D., Megowan-romanowicz, C., Birchfield, D. and Megowan-romanowicz, C. 2009. Earth Science Learning in SMALLab: a Design Experiment for Mixed-Reality. *Journal of Computer Supported Collaborative Learning*. 4, 4 (2009), 403–421.
- [5] Bosch, N. and Paquette, L. 2021. What’s Next? Sequence Length and Impossible Loops in State Transition Measurement. 13, 1 (2021), 23.
- [6] Bruner, J. 1985. The Role of Interaction Formats in Language Acquisition. *Language and Social Situations*. J.P. Forgas, ed. Springer. 31–46.
- [7] Chang, C.-J., Chang, M.-H., Chiu, B.-C., Liu, C.-C., Fan Chiang, S.-H., Wen, C.-T., Hwang, F.-K., Wu, Y.-T., Chao, P.-Y., Lai, C.-H., Wu, S.-W., Chang, C.-K. and Chen, W. 2017. An analysis of student collaborative problem solving activities mediated by collaborative simulations. *Computers & Education*. 114, (Nov. 2017), 222–235. DOI:<https://doi.org/10.1016/j.compedu.2017.07.008>.
- [8] D’Angelo, S. and Schneider, B. 2021. Shared Gaze Visualizations in Collaborative Interactions: Past, Present and Future. *Interacting with Computers*. 33, 2 (Mar. 2021), 115–133. DOI:<https://doi.org/10.1093/iwcomp/iwab015>.
- [9] Dede, C. 2009. Immersive Interfaces for Engagement and Learning. *Science*. 323, 5910 (Jan. 2009), 66–69. DOI:<https://doi.org/10.1126/science.1167311>.
- [10] Diederich, M., Kang, J., Kim, T. and Lindgren, R. 2021. Developing an In-Application Shared View Metric to Capture Collaborative Learning in a Multi-Platform Astronomy Simulation. *LAK21: 11th International Learning Analytics and Knowledge Conference* (Irvine CA USA, Apr. 2021), 173–183.
- [11] Dunleavy, M., Dede, C. and Mitchell, R. 2009. Affordances and Limitations of Immersive Participatory Augmented Reality Simulations for Teaching and Learning. *Journal of Science Education and Technology*. 18, 1 (Feb. 2009), 7–22. DOI:<https://doi.org/10.1007/s10956-008-9119-1>.
- [12] Emara, M., Hutchins, N., Grover, S., Snyder, C. and Biswas, G. 2021. Examining Student Regulation of Collaborative, Computational, Problem-Solving Processes in Open-Ended Learning Environments. *Journal of Learning Analytics*. 8, 1 (Apr. 2021), 49–74. DOI:<https://doi.org/10.18608/jla.2021.7230>.
- [13] Gabadinho, A., Studer, M., Müller, N., Bürgin, R., Fonta, P.-A. and Ritschard, G. 2022. *TraMineR: Trajectory Miner: a Toolbox for Exploring and Rendering Sequences*.
- [14] Griffin, P. and Care, E. Educational Assessment in an Information Age. 282.
- [15] Han, A., Krieger, F. and Greiff, S. 2021. Collaboration Analytics Need More Comprehensive Models and Methods: An Opinion Paper. *Journal of Learning Analytics*. 8, 1 (Apr. 2021), 13–29. DOI:<https://doi.org/10.18608/jla.2021.7288>.
- [16] Helske, J. and Helske, S. 2021. *seqHMM: Mixture Hidden Markov Models for Social Sequence Data and Other Multivariate, Multichannel Categorical Time Series*.
- [17] Eduardo, H., Radu, I., Joy, T. and Schneider, B. 2021. Augmented Reality in Collaborative Problem Solving: A Qualitative Study of Challenges and Solutions. *International Conference on Computer Supported Collaborative Learning* (2021).
- [18] Jermann, P., Mullins, D., Nüssli, M.-A., Dillenbourg, P. and Nuessli, M.-A. 2011. Collaborative Gaze Footprints: Correlates of Interaction Quality. (2011), 8.
- [19] Kang, J., An, D., Yan, L. and Liu, M. Collaborative Problem-Solving Process in A Science Serious Game: Exploring Group Action Similarity Trajectory. (2019), 6.
- [20] Lämsä, J., Hämäläinen, R., Koskinen, P., Viiri, J. and Mannonen, J. 2020. The potential of temporal analysis: Combining log data and lag sequential analysis to investigate temporal differences between scaffolded and non-scaffolded group inquiry-based learning processes. *Computers & Education*. 143, (Jan. 2020), 103674. DOI:<https://doi.org/10.1016/j.compedu.2019.103674>.
- [21] Lin, T.-J., Duh, H.B.-L., Li, N., Wang, H.-Y. and Tsai, C.-C. 2013. An investigation of learners’ collaborative knowledge construction performances and behavior patterns in an augmented reality simulation system. *Computers & Education*. 68, (Oct. 2013), 314–321. DOI:<https://doi.org/10.1016/j.compedu.2013.05.011>.
- [22] Martinez-Maldonado, R., Yacef, K., Kay, J., Kharrufa, A. and Al-Qaraghuli, A. Analysing frequent sequential patterns of collaborative learning activity around an interactive tabletop. *International Conference on Educational Data Mining (EDM 2011)* (2011), 111–120.
- [23] Matayoshi, J. and Karumbaiah, S. 2020. Adjusting the L Statistic when Self-Transitions are Excluded in Affect Dynamics. 12, 4 (2020), 23.
- [24] Mercier, E.M. and Higgins, S.E. 2013. Collaborative learning with multi-touch technology: Developing adaptive expertise. *Learning and Instruction*. 25, (Jun. 2013), 13–23. DOI:<https://doi.org/10.1016/j.learninstruc.2012.10.004>.
- [25] Moore, C. and Dunham, P.J. 1995. *Joint attention: its origins and role in development*. Lawrence Erlbaum Associates.
- [26] OECD 2017. *Development Co-operation Report 2017: Data for Development*. Organisation for Economic Co-operation and Development.
- [27] O’Madagain, C. and Tomasello, M. 2021. Joint attention to mental content and the social origin of reasoning. *Synthese*. 198, 5 (May 2021), 4057–4078. DOI:<https://doi.org/10.1007/s11229-019-02327-1>.
- [28] Phelps, E., & Damon, W. (1989). Problem solving with equals: Peer collaboration as a context for learning mathematics and spatial concepts. *Journal of Educational Psychology*, 81(4), 639–646.
- [29] Qvarfordt, P., Beymer, D. and Zhai, S. 2005. RealTourist – A Study of Augmenting Human-Human and Human-Computer Dialogue with Eye-Gaze Overlay. *Human-Computer Interaction - INTERACT 2005*. M.F. Costabile and F. Paternò, eds. Springer Berlin Heidelberg. 767–780.

- [30] Richardson, D.C. and Dale, R. 2005. Looking To Understand: The Coupling Between Speakers' and Listeners' Eye Movements and Its Relationship to Discourse Comprehension. *Cognitive Science*. 29, 6 (Nov. 2005), 1045–1060. DOI:https://doi.org/10.1207/s15516709cog0000_29.
- [31] Rodríguez, F.J. and Boyer, K.E. 2015. Discovering Individual and Collaborative Problem-Solving Modes with Hidden Markov Models. *Artificial Intelligence in Education*. C. Conati, N. Heffernan, A. Mitrovic, and M.F. Verdejo, eds. Springer International Publishing. 408–418.
- [32] Roschelle, J. and Teasley, S.D. 1995. The Construction of Shared Knowledge in Collaborative Problem Solving. *Computer Supported Collaborative Learning*. C. O'Malley, ed. Springer Berlin Heidelberg. 69–97.
- [33] Schneider, B. and Pea, R. 2013. Real-time mutual gaze perception enhances collaborative learning and collaboration quality. *International Journal of Computer-Supported Collaborative Learning*. 8, 4 (Dec. 2013), 375–397. DOI:<https://doi.org/10.1007/s11412-013-9181-4>.
- [34] Schneider, B., Sharma, K., Cuendet, S., Zufferey, G., Dillenbourg, P. and Pea, R. 2018. Leveraging mobile eye-trackers to capture joint visual attention in co-located collaborative learning groups. *International Journal of Computer-Supported Collaborative Learning*. 13, 3 (Sep. 2018), 241–261. DOI:<https://doi.org/10.1007/s11412-018-9281-2>.
- [35] Shaer, O., Strait, M., Valdes, C., Feng, T., Lintz, M. and Wang, H. 2011. Enhancing genomic learning through tabletop interaction. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Vancouver BC Canada, May 2011), 2817–2826.
- [36] Siposova, B. and Carpenter, M. 2019. A new look at joint attention and common knowledge. *Cognition*. 189, (Aug. 2019), 260–274. DOI:<https://doi.org/10.1016/j.cognition.2019.03.019>.
- [37] Wang, H.-Y., Duh, H., Li, N., Lin, T.-J. and Tsai, C.-C. 2014. An Investigation of University Students' Collaborative Inquiry Learning Behaviors in an Augmented Reality Simulation and a Traditional Simulation. *Journal of Science Education and Technology*. 23, 5 (2014), 682–691. DOI:<https://doi.org/10.1007/s10956-014-9494-8>.
- [38] Wayntal, D., Serna, A., Pernelle, P. and marty, jean-charles 2015. Multi-Devices Territoriality to Manage Collaborative Activities in a Learning Game. *9th European Conference on Games-Based Learning (ECGBL 2015)* (Steinkjer, Norway, Oct. 2015), 570–578.
- [39] Wu, H.-K., Lee, S.W.-Y., Chang, H.-Y. and Liang, J.-C. 2013. Current status, opportunities and challenges of augmented reality in education. *Computers & Education*. 62, (Mar. 2013), 41–49. DOI:<https://doi.org/10.1016/j.compedu.2012.10.024>.
- [40] Zheng, J., Xing, W. and Zhu, G. 2019. Examining sequential patterns of self- and socially shared regulation of STEM learning in a CSCL environment. *Computers & Education*.

Can Population-based Engagement Improve Personalisation? A Novel Dataset and Experiments

Sahan Bulathwela¹, Meghana Verma², María Pérez-Ortiz¹, Emine Yilmaz¹ and John Shawe-Taylor¹

¹ Centre for Artificial Intelligence, University College London, UK

² Indian Institute of Technology Bombay, India
m.bulathwela@ucl.ac.uk

ABSTRACT

This work explores how population-based engagement prediction can address cold-start at scale in large learning resource collections. The paper introduces i) VLE, a novel dataset that consists of content and video based features extracted from publicly available scientific video lectures coupled with implicit and explicit signals related to learner engagement, ii) two standard tasks related to predicting and ranking context-agnostic engagement in video lectures with preliminary baselines and iii) a set of experiments that validate the usefulness of the proposed dataset. Our experimental results indicate that the newly proposed VLE dataset leads to building context-agnostic engagement prediction models that are significantly performant than ones based on previous datasets, mainly attributing to the increase of training examples. VLE dataset's suitability in building models towards Computer Science/ Artificial Intelligence education focused on e-learning/ MOOC use-cases is also evidenced. Further experiments in combining the built model with a personalising algorithm show promising improvements in addressing the cold-start problem encountered in educational recommenders. This is the largest and most diverse publicly available dataset to our knowledge that deals with learner engagement prediction tasks. The dataset, helper tools, descriptive statistics and example code snippets are available publicly.

Keywords

Population-based Engagement, Cold-start, Educational Recommender, Personalised Education, AI in Education

1. INTRODUCTION

With the growth of Open Educational Resources (OER) [38, 6, 32] and Massively Open Online Courses (MOOC) [34, 23], large educational repositories need scalable tools to understand the engagement potential of newly added materials [14]. While *contextualised engagement* can be defined as learner's engagement driven by the context at a given time

in their learning path (e.g., learning needs/goals, knowledge state etc.), *context-agnostic engagement* aims to capture patterns and features associated with engagement that instead are applicable to an entire learner population rather than individual contexts of specific learners [7]. Put simply, context-agnostic engagement is concerned with the features that generally make an educational material engaging.

While many datasets capturing contextual engagement of learners exist, our contribution: **Video Lecture Engagement (VLE)**, focusing on context-agnostic engagement, is a novel dataset that presents around 12,000 peer-reviewed scientific videos constructed from a popular OER repository and contains a variety of lecture types ranging from scientific talks and expert panels to MOOC-like lectures. VLE provides textual and video-specific features extracted from the lecture transcripts, together with Wikipedia topics covered in the video (via entity linking) and user engagement labels (both explicit and implicit) for each video. In educational recommenders, VLE dataset helps solving both i) *user cold-start*, where new users join the system and we may not have enough information about their context and ii) *item cold-start*, where new educational content is released, for which we may not have user engagement data yet and thus an engagement predictive model would be necessary. While utility of context-agnostic engagement models for cold start is previously demonstrated [7], VLE is the largest dataset for building such models. This work is aimed not at replacing personalised recommendation but to complement it by addressing the cold-start problem. While VLE dataset is a major contribution of this work, several additional experimental results make up the overall contribution. These results demonstrate the usefulness of VLE dataset via answering a set of critical research questions.

2. RELATED WORK

The majority of work in Intelligent Tutoring Systems (ITS) and Educational Recommendation Systems (EduRecSys) revolve around contextual learner engagement [26, 9]. While many explore the connection between engagement and learning gains [2, 20, 28], public datasets in this realm are hard to come by. MOOC platforms such as edX [23] and Khan Academy [29] harvest valuable data created in an *in-the-wild* setting, yet this data is gated within course owners and consortia [21] (or heavily anonymised) due to its proprietary nature. However, with the boom of online education, it is greatly imperative that such datasets are democratised so that under-researched areas such as context-agnostic

S. Bulathwela, M. Verma, M. P. Ortiz, E. Yilmaz, and J. Shawe-Taylor. Can population-based engagement improve personalisation? A novel dataset and experiments. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 414–421, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853185>

(population-based) engagement can be widely understood to push the frontiers of AI-supported online education. Study of context-agnostic engagement of video lectures so far has been mainly qualitative, deriving guidelines such as keeping videos short and in parts [23, 3]. While these findings are useful in content creation stage, they have little use in moderating the mammoth of materials already circulating in the Internet. Our work proposes building predictive models that can be utilised post-creation for scalable quality assurance and content recommendation.

2.1 Related Datasets

Many video engagement prediction works revolve around YouTube [16, 24] and uses platform specific features (e.g. channel reputation, category etc.) exclusively. While large-scale datasets for this task is published, these datasets include general-purpose videos (largely entertainment related) rather than educational videos [40]. Some features proposed in these works share similarity to our proposal (such as duration, language and topic features). However, no content-based features (based on transcript) relating to understandability and presentation are used, making the methods hard to generalise outside of YouTube. Educational Information Retrieval [36, 15] and Wikipedia page quality prediction [19, 39] has been attempted using features such as text style, readability, structure, network, recency and review data. Publicly available Wikipedia article quality dataset [19] with human annotated (explicit) labels is used to tackle the latter task although implicit labels are not included in this dataset. Similar datasets are available for automated essay scoring [37]. But, none of these datasets fill the lack of datasets for predicting engagement of educational videos.

In the educational context, measuring learner engagement using multi-modal data such as brain waves [41], facial expressions [25] etc. are conducted in controlled environments and lacks "in-the-wild" signals [22]. Large public datasets and competitions also relate to students answering questions in-the-wild (e.g. ASSISTments [30] or multiple choice questions [13]), contrary to the proposed VLE dataset, lacks implicit feedback relating to learners acquiring skills/knowledge. More relevant datasets studying population-based engagement revolves around MOOCs. Studying approximately 800 videos from edX platform, Guo et al. [23] manually processed and provided a qualitative analysis of engagement, with a few features being relatively subjective and difficult to automate. A similar work [35] takes 22 edX videos, extracts cross-modal features and manually annotates their quality with no focus on learner engagement. Neither dataset is publicly available. MOOCcube is a recently released dataset that contains a spectrum of details relating to MOOC interactions [42]. Although large, the video watch logs in MOOCcube come from 190,000 users contrary to over 1.1 Million users of VLE. The dataset is also not tested for engagement prediction by its publishers raising uncertainty in its promise.

Our prior work [7] explored the possibility of building context-agnostic engagement models using implicit watch time-based labels and showed that these models can address the cold-start problem. We identify this contribution most relevant to the proposed dataset. Our prior work publishes a dataset of 4,000 lectures with labels coming from 150,000 learners.

VLE, expands this dataset with 3 times as many videos with engagement signals generated by 7 times as many learners. The new dataset also restricts itself to English lectures to refine relevance of the proposed features. Additionally, VLE introduces a new set of Wikipedia-based topical features.

3. VLE DATASET

VLE dataset is created using the aggregated video lectures consumption data coming from VideoLectures.Net (VLN). These videos are recorded when researchers are presenting their work at peer-reviewed conferences. Lectures are thus reviewed and material is controlled for correctness of knowledge. The collection consists of scientific talks and tutorials that are mainly geared towards university level learners. In that aspect, many videos in the dataset are stylistically similar to conventional MOOC lectures. VLE dataset provides a set of features together with labels based on subjective assessment metrics such as star ratings and view count. We believe that this dataset enables understanding the connection between content features and the collective engageability of learners with an educational video.

3.1 Feature Extraction

The video metadata and transcriptions are transformed into i) content-based textual features, ii) Wikipedia topic-based features and iii) video-specific features. Majority of the extracted features are cross modal (e.g. books, websites and audios) and are easily automatable.

Content-based Features. Based on prior proposals [7], we extract *Word Count* [39], *Title Word Count* and *Document Entropy* [1], language style features [18], *Preposition Rate*, *Auxiliary Rate*, *To Be Rate*, *Conjunction Rate*, *Normalisation Rate*, *Pronoun Rate*, readability related *Easiness (FK Easiness)* [18] and vocabulary related *Stop-word Presence Rate*, *Stop-word Coverage Rate* [1, 31]. To represent *Freshness* of lectures (recency), we calculate the number of days between January 01, 1960 and the video published date to use it as a proxy for recency of the lecture [7].

Wikipedia-based Features. We use Wikifier [4] to extract topical features capturing topic authority and coverage.

The *top-5 authoritative topic URLs* and *top-5 PageRank scores* features represent the Topic Authority feature vertical. Wikifier [4] produces a PageRank score [5] that indicates the marginal authoritative of a Wikipedia concept among all Wikipedia concepts associated with a lecture. It is noteworthy that *authority* of a learning resource entails author, organisation and content authority [11]. The proposed features represent content authority.

The *top-5 covered topic URLs* and *top-5 cosine similarity scores* features represent *Topic Coverage* feature vertical. The cosine similarity score $\cos(s_{tr}, c)$ between the *Term Frequency-Inverse Document Frequency (TF-IDF)* representations of the lecture transcript s_{tr} and the Wikipedia page of concept c is also an output from the Wikifier. These features are used as a proxy for topic coverage.

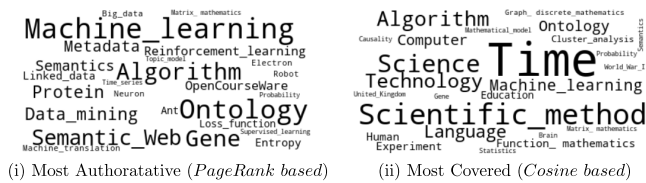


Figure 1: WordClouds summarising the distribution of top 25 (i) most authoritative and (ii) most covered Wikipedia topics in the dataset. Note that Data Science and Computer Science related topics are most dominant topics.

These four feature sets create 20 distinct feature columns. Figure 1 presents two word clouds that show the 25 most authoritative and covered topics in the VLE dataset and show that there are distinct differences between topic presence in the two feature sets. The authoritative topics are topics that are highly connected to other topics in the lecture whereas covered topics have high textual overlap between lecture transcript and Wikipedia concept page.

Video-specific Features. We identify a set of easily automatable, prior proposed [7] video specific features. *Lecture Duration*, *Is Chunked*, *Lecture Type*[23], *Silence Period Rate (SPR)* and *Speaker Speed* [7] are calculated based on prior work. *Lecture Duration* is reported in seconds. *Is Chunked* is a binary feature which indicates if a lecture has multiple parts. *Lecture type* value is derived from the metadata. This diverse dataset contains many different types of videos ranging from presentations, panels to tutorials.

3.2 Labels

Three main types of quantification of engagement labels are presented in the dataset.

Explicit Ratings and Popularity. *Mean Star Rating* based on a scale of 1-5 (5 being best) for each video is provided. Ratings are paired with the number of ratings used to compute the mean. The proposed dataset has 2,127 ratings (almost 2x than [7]). Missing ratings are labelled with -1. Capturing popularity, the total number of views, named *View Count*, for each video as of February 1, 2021 is provided.

Watch Time/Engagement. Many labels in this dataset are based on *watch time* [16]. Normalised Engagement Time (NET) is computed for each video as it has been proposed as the gold standard for learner engagement [23]. The Median of NET (MNET) and Average of NET (ANET) is calculated from NET. To have the MNET and ANET labels in the range [0, 1], we set the upper bound to 1, deriving Saturated MNET (*SMNET*) and Saturated ANET (*SANET*) that are included in the dataset. The standard deviation of NET (*Std of Engagement*) is also reported, together with the *Number of User Sessions* used for calculating MNET and ANET. These measures allow understanding stability of the centres published. The set of individual NET values are also published to allow future researchers to exploit the true distribution of values.

3.3 Preserving Anonymity and Ethics

Users of VLN repository formally agree that all user generated content is available for research. We further anonymise and aggregate interactions to ensure privacy. Aggregated results are only published in lectures with atleast 5 user views to preserve k-anonymity [17]. The presenters and authors of videos in VLN also formally agree to the video, presentation slides and supplementary material to be published under an open license and can be used for educational and research purposes. Still, a regime of techniques outlined below are used to preserve presenter anonymity in order to avoid having unanticipated effects on lecturer’s reputation by associating implicit learner engagement values to their content. Rarely occurring *Lecture Type* values are grouped together to create the **other** category. Life Sciences, Physics, Technology, Mathematics, Computer Science, Data Science and Computers categories are grouped as **stem** and all other categories as **misc** category. Rounding is used with *Freshness* and *Lecture Duration* to the nearest 10 days and 10 seconds respectively. Gaussian white noise (10%) is added to *Title Word Count* feature and rounded to the nearest integer.

VLN repository is concentrated with videos about Computer Science (see Figure 1), a subject area with gender imbalance in both audience and presenters. We avoided using feature classes that could potentially reflect gender characteristics to improve neutrality of the dataset (and models). Visual features (facial features, emotions...) and audio features (pitch, tone...) that may actively/passively embed gender is avoided. We focused primarily on features that reflect informational content. Where video specific features are used, generic features such as "speaker speed" that are unlikely to be correlated to gender or age are used.

3.4 Final Dataset

The final dataset contains 11,548 lectures across 21 subjects (eg. Computer Science, Philosophy, etc. with a majority from AI and Computer Science) that are published between September 1, 1999 and December 31, 2020. The engagement labels are created from events of over 1.1 Million users logged between December 01, 2016 and February 01, 2020. The collection of videos span various video lengths with the duration distribution having two modes at approx. 2000s (33 mins) and 4000s (1hr) time points which align with typical lengths of research talks and presentations. The mean word count of the videos is 5347.9. The video lecture collection uses on average 93.9 learners per video when calculating engagement centres. The dataset, helper tools and example code snippets are available publicly¹.

3.5 Supported Tasks

Scalable Quality Assurance and Educational Recommendation are two key downstream applications of context-agnostic engagement prediction. We establish two main tasks, which we mainly focus on in this paper, that can be objectively addressed using the VLE dataset. These are:

- **Task 1:** Predicting context-agnostic (population-based) engagement of video lectures
- **Task 2:** Ranking of video lectures based on engagement

¹<https://github.com/sahanbull/VLE-Dataset>

Other Tasks. Beyond the proposed tasks, this dataset is suitable for, not limiting to, several tasks such as i) understanding influential features for engagement prediction and ii) understanding the strengths and weaknesses of different implicit/explicit labels, that have been investigated in our prior work with similar datasets [7, 33]. The textual and topical representations, with the use of unsupervised approaches can be used in a range of tasks from understanding meaningful hidden patterns within clusters of videos (e.g. talks vs. lectures vs. tutorials) to deducing the structure of knowledge based on how topics co-occur within videos.

3.6 Evaluating Performance

We identify *Root Mean Squared Error (RMSE)* as a suitable metric for evaluating Task 1. Measuring RMSE against the original labels published with the datasets will allow different works to be compared fairly. With reference to Task 2, we identify *Spearman's Rank Order Correlation Coefficient (SROCC)* as a suitable metric. SROCC is suitable for comparing between ranking models that create global rankings (e.g. point-wise rankers).

We use 5-fold cross validation to evaluate model performance with tasks 1 and 2. The folds are released together with the dataset, to allow to facilitate fair comparison and reproducibility. 5-fold cross validation allows reporting the *confidence intervals* ($1.96 \times \text{Standard Error}$) of the performance estimate, which we include in Table 1.

4. BASELINES AND EXPERIMENTS

Through our experiments, we aim to answer research questions relevant to the supported tasks (in section 3.5) and other facets that further demonstrate the utility of this dataset. The main research questions of interest are:

- **RQ1:** Does the newly constructed VLE dataset lead to training more performant prediction models?
- **RQ2:** How does the larger quantity of training data affect predictive performance?
- **RQ3:** Is the model useful for modelling engagement with Computer Science materials?
- **RQ4:** Is this dataset useful for modelling engagement in E-Learning lectures and MOOC videos?
- **RQ5:** Does context-agnostic engagement prediction help in the cold-start scenario?

Our previous work [7] demonstrated *Random Forest (RF)* model obtains best performance among linear and non-linear models in similar datasets. Therefore, we use the RF model to benchmark the new VLE dataset for Tasks 1 and 2 described earlier.

4.1 Labels and Features for Baseline Models

SMNET label is used as the target variable for both tasks. Preliminary investigations indicated that SMNET label follows a Log-Normal distribution, motivating us to use a log transformation on the SMNET values before training the models. Empirical results further confirmed that this step

improves the final performance of the models. We undo this transformation for computing *RMSE* while this transformation doesn't affect *SROCC*.

All the features outlined as the content-based and video-based sections in section 3.1 are included in the baseline models. The models are trained with three different feature sets in an incremental fashion:

1. *Content-based:* Features extracted from lecture metadata and the transcript-based textual features.
2. *+ Wiki-based:* Content-based + 2 Wikipedia-based features (Top 1 Most Authoritative Topic URL and Most Covered Topic URL).
3. *+ Video-based:* Content-based + Wiki-based + Video-specific features.

However, due to the large amount of topics in the Wikipedia-based feature groups, we restrict to the top 1 authoritative and covered topic features where they are encoded as binary categorical variables. Practitioners are encouraged to try further encoding of the topic variables, as it will likely have a positive impact on the performance.

4.2 Experiments

Addressing RQ1, the RF models are trained with the three proposed feature sets using 5-fold cross validation with the prior proposed, smaller 4k dataset [7] and the newly proposed VLE dataset (12k). This setup allows identifying how performance gains are achieved through i) adding each new group of features and ii) adding new observations. Follow on experiments addressing RQ2 and RQ3 are run using hold-out validation technique where fold 5 is held out. We experiment by using varying proportions of training data in RQ2 to train the model. When selecting training data, random sampling is used. All the trained models in RQ2 are evaluated using the same held out test set.

To validate RQ4, we partition the entire dataset into i) tutorial videos (`vtt` lecture type) and ii) all other videos, as test and train data respectively. However, tutorials conducted in conferences significantly vary from e-learning videos geared for MOOCs. To address this mismatch, we further identified 1,035 videos (among the tutorials) that exclusively belong to the Open Course Ware Consortium (OCWC)². OCWC exclusively contains university lectures intended for teaching and devises different MOOC production techniques such as classroom, talking head and power point methods [23]. In RQ5, we train the model using all non-tutorial lectures and test the engagement prediction/ranking performance on i) OCWC videos (`ocw`), ii) all tutorials but OCWC (`!ocw`) and iii) all tutorials `vtt`, (entire test set). Experiments of (RQ2-4) are only done with the best performing model from Table 1 (RF model with *Content + Wiki + Video* feature group) to reduce computational cost.

To tackle RQ5, we utilise TrueLearn Novel [9] (hereby referred to as *TrueLearn*), a personalisation model that predicts learner engagement with video lectures. A key limitation of many such models is lack of information in the

²<http://videolectures.net/ocwc>

Table 1: RMSE and SROCC with confidence intervals for the engagement prediction (Task 1) and lecture ranking (Task 2) using the Random Forests model with both 4k [7] and 12k (Our VLE) datasets. Better performance highlighted in bold.

Feature set	RMSE with Task 1		SROCC with Task 2	
	4k	12k (<i>Ours</i>)	4k	12k (<i>Ours</i>)
Content-based	.1801±.006	.1170±.006	.6190±.011	.7504±.013
+ Wiki-based	.1798±.007	.1178±.006	.6251±.014	.7505±.013
+ Video-specific	.1728±.007	.1098±.007	.6758±.020	.7832±.009

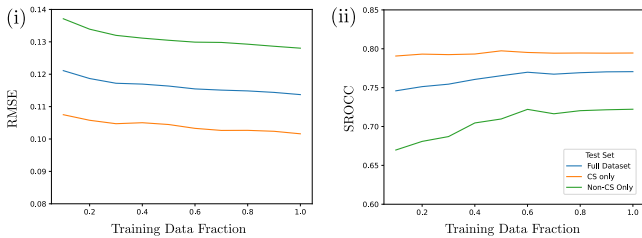


Figure 2: Predictive performance for (i) engagement prediction and (ii) lecture ranking tasks with varying proportions of randomly sampled training data. The test set performance for full test dataset (Blue) and subsets of test dataset that consists of CS lectures only (Orange) and Non-CS lectures only (Green) are also reported

Table 2: Performance for OpenCourseWare (ocw), Non-OpenCourseWare (!ocw) tutorial and All tutorial (vtt) videos for engagement prediction and lecture ranking tasks. Better performance per task is highlighted in bold.

	ocw	!ocw	vtt	From Table 1
RMSE with Task 1	.0539	.0404	.0406	.1098
SROCC with Task 2	.9485	.9209	.9223	.7832

early stages of the user session (*cold-start* problem) to make accurate predictions. In this experiment, we created a hybrid recommender that combines the TrueLearn model with the proposed context-agnostic engagement prediction model (hereby referred to as *TrueLearn++*). For simplicity, we use "switching" [12] in *TrueLearn++*, where the pre-trained context-agnostic model makes the prediction on the *first* event of each user (where the personalisation model has no information) and switches to TrueLearn that can exploit user history. PEEK dataset [8], with more than 20,000 learner sessions, is used for the experiment where the context agnostic model is trained using lectures not present in the PEEK test data. Then the predictive performance on the PEEK test data using TrueLearn (The baseline) and *TrueLearn++* (the hybrid model) is measured using Accuracy, Precision, Recall and F1-Score. A learner-wise, one-tailed paired t-test is used for statistical significance testing.

5. RESULTS AND DISCUSSION

The performance metrics observed with the RF model on Task 1 and 2 (RQ1) are outlined in Table 1. Figure 2 illustrates how the training data size impacts the i) RMSE and ii) SROCC (RQ2) as well as showing the engagement prediction performance on Computer Science (CS) videos vs. Non-CS videos (RQ3). Table 2 presents predictive performance of the model on e-learning type lectures and tutorials (RQ4). Finally, the overall performance comparison relating

to the effect of *combining* the context agnostic model with personalisation models to battle cold-start problem (RQ5) is reported in Table 3.

5.1 Performance Gains and Causes (RQ1-2)

Table 1 shows that the larger VLE leads to significant performance gains in both engagement prediction and video ranking tasks over the 4k dataset [7]. When using all feature sets with the RF RMSE on Task 1 drops by 41% while SROCC on Task 2 jumps by 15%. The labels in VLE dataset have more statistical stability as the centres are computed using more data points collected over a wider time window. Using more feature groups also leads to improved performance. Results for VLE dataset in Table 1 shows this trend where best performance is evidenced with all the features (+ *Video-specific* group) used. However, using cross-modal content-based features (*Content-based* + *Wiki-based*) alone leads to substantial performance. This result indicates that engagement prediction is still feasible only depending on easy to automate, cross modal features. Wikification, used in generating Wiki-based features, also operates in web-scale³. While Table 1 results don't show a significant bump when Wiki-based features are added, we believe that this is due to the simplicity of features used with much room for more sophisticated features (e.g. semantic relatedness between Wikipedia topics [10]) that will lead to performance gains.

Figure 2 confirms that the increase of training data improves performance in both tasks. RMSE continues to shrink in Figure 2(i) while SROCC in Figure 2(ii) tells a different story where improvements saturate at 60%. This suggests that improving ranks gets significantly harder around 5,500 training examples ($\approx 60\%$ of training set).

5.2 Relevance to AI/CS Education (RQ3) and E-learning Scenarios (RQ4)

Figure 2 shows that VLE dataset is suitable for training models for CS-only lectures leading to test set RMSE of $\approx .1$ and SROCC of $\approx .8$. This may be due to i) the higher diversity (significant differences between subjects) of lectures within the non-CS group and ii) the majority of CS/AI (e.g. Machine Learning, Ontology, Semantic Web etc.) related videos being present within the VLE dataset (see Figure 1). Table 2 shows strong evidence that the models trained with VLE dataset generalise really well for engagement modelling in e-learning type videos created for course teaching amid the dataset containing many different video types. The models trained are much better at engagement prediction and ranking of tutorial-like videos than general scientific talks. Having tested with lectures that have been recorded using different MOOC video production techniques, the high performance obtained on *ocw* lectures confirms that VLE dataset can be highly effective in building context-agnostic engagement models for e-learning and MOOC systems.

5.3 Addressing the Cold-Start Problem (RQ5)

Table 3 shows that simply combining the context-agnostic engagement prediction with TrueLearn Novel algorithm (together becoming *TrueLearn++*) can lead to significant improvements in accuracy and precision. The same table also

³<http://wikifier.org>

Table 3: Average test set performance for Accuracy (Acc.), Precision (Prec.), Recall (Rec.) and F1-Score (F1) in predicting *first* event and *all* events. The more performant value is highlighted in bold. The metrics where the proposed model that outperform the baseline counterpart in the PEEK dataset ($p < 0.01$ in a one-tailed paired t-test) are marked with $\cdot^{(*)}$.

Model	Predicting <i>first</i> event				Predicting <i>all</i> events			
	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1
Truelearn	44.21	44.21	100.00	61.32	62.69	57.54	81.88	64.98
Truelearn ₊₊	56.09 $\cdot^{(*)}$	50.32 $\cdot^{(*)}$	53.58	51.90	63.51 $\cdot^{(*)}$	57.91 $\cdot^{(*)}$	79.13	64.39

shows that the drop of overall F1 score can be attributed to the comparatively steeper drop of Recall Score. Inspecting left part of Table 3 sheds more light into where this steep drop of recall occurs. This is, the TrueLearn model always predicts positive engagement for the first event of the user. At the *first* event, the recall of TrueLearn being 1.0 while the accuracy and precision being the same depicts this fact. TrueLearn predicts positive in event 1 of each user because the model has no information to base the prediction on [9]. However, the scenario is different in TrueLearn₊₊ in the first event as the model has additional information. Both accuracy and precision of predictions in first event of the learner population significantly improves. The recall will fall as the proposed context-agnostic model only captures a population based prior which may deviate from the individuality of the learners. However, it can be argued that making a prediction with additional information is better than predicting with no prior information. In the bigger picture (in Table 3 right part), being able to make slightly more informed and varied predictions for the first event of learners based on lecture content features enable significantly improving prediction accuracy and precision of TrueLearn₊₊. It is also noteworthy that our experiment, for the sake of simplicity, uses a rule that could be significantly improved further, e.g. using weights of the probabilities of both population-based and personalised models at the beginning of a user session (also known as stacking [12]), where the weight of population-based engagement decreases as we gather more information about the user.

5.4 Opportunities and Limitations

The VLE dataset is one of the biggest datasets for modelling context-agnostic engagement of educational videos (15 \times than [23] and 3 \times than [7]). This unlocks potential to apply complex model families (e.g. deep learning) with the potential to periodically expand the dataset to further push the frontiers of research. The Wiki-based features open up limitless possibilities as many sophisticated feature sets can be built and experimented. Due to the connectivity to Wikipedia, both its content and link structures can be exploited to invent meaningful, yet interpretable features. A further step can enable other data structures such as knowledge bases (e.g Wikidata) and category trees. to be used for feature creation. As the VLE dataset captures how content features relate to engagement, this dataset can be used to solidify our understanding on how to create engaging learning materials [27, 23, 3, 7].

There also exists limitations. VLE dataset is largely comprised of Computer Science and Data Science materials (Figure 1) that are delivered all in English. While this is an opportunity for AI and Computer Science education, results in Figure 2 also shows that this fact leads to comparatively

less fruitful non-CS results. The dataset and its features are also not suitable for non-English video collections. Amid its size, the dataset still lacks variety of materials in topical and lingual sense. As pointed out in section 3.3, we have taken some measures to restrict the feature set to what we believe to be more neutral features that do not discriminate gender or age. However, since we do not have access to gender information in the data collected, it is impossible to test and guarantee that VLE dataset doesn't carry negative gender, age biases. Care should be taken when enhancing these features and there is room to do more rigorous tests to understand if any gender biases are present within the dataset. *Learner Engagement* is a loaded concept with many facets. In relation to consuming videos, many behavioural actions such as pausing, rewinding and skipping can contribute to latent engagement with a video lecture [28]. Due to the technical limitations of the platform and privacy concerns, only watch time, view and mean ratings are included in this dataset. Although watch time has been used as a representative proxy for learner engagement with videos [23, 40, 16], we acknowledge that more informative measures may lead to more complete engagement signals.

6. CONCLUSION

In order to push the frontiers of context-agnostic engagement prediction, we construct and release the VLE dataset consisting of i) content-based, ii) Wiki-based and iii) video-specific features with multiple explicit and implicit labels. Two formal tasks are established to predict engagement and rank videos with baseline models that significantly outperform predecessors. Empirically, i) improvement of performance with training data size, ii) suitability of VLE dataset for CS/AI education, iii) relevance to MOOCs and e-learning and iv) the feasibility of using context-agnostic engagement prediction to address cold-start problem in personalised (contextual) educational recommendation is demonstrated.

In retrospect of section 5.4, expanding the dataset vertically (with diverse observations) and horizontally (with novel features) is our highest priority going forward. As a future direction, much richer learner engagement signals (e.g. pauses, replays, skips etc.) can be incorporated to the dataset without compromising user privacy. When better understanding of learner engagement is gained, training examples coming from other modalities (e.g. PDF and E-books) can be added to the dataset to further widen the scope of the dataset, enabling understanding of learner engagement across different types of learning resources. The attempt to improve personalisation using population-based models in our work is barely scratching the surface. There is a lot of potential to extensively propose sophisticated and rigorously tested approaches to exploit this idea, which we will also explore in future work.

7. ACKNOWLEDGMENTS

This research was partially conducted as part of the X5GON project funded from the EU's Horizon 2020 research programme grant No 761758. This work is also supported by the European Commission funded project "Humane AI: Toward AI Systems That Augment and Empower Humans by Understanding Us, our Society and the World Around Us" (grant 820437) and the EPSRC Fellowship titled "Task Based Information Retrieval" (grant EP/P024289/1).

8. REFERENCES

- [1] M. Bendersky, W. B. Croft, and Y. Diao. Quality-biased ranking of web documents. In *Proc. of ACM Int. Conf. on Web Search and Data Mining*, 2011.
- [2] F. Bonafini, C. Chae, E. Park, and K. Jablokow. How much does student engagement with videos and forums in a mooc affect their achievement? *Online Learning Journal*, 21(4), 2017.
- [3] C. J. Brame. Effective educational videos: Principles and guidelines for maximizing student learning from video content. *CBE—Life Sciences Education*, 15(4), 2016.
- [4] J. Brank, G. Leban, and M. Grobelnik. Annotating documents with relevant wikipedia concepts. In *Proc. of Slovenian KDD Conf. on Data Mining and Data Warehouses (SiKDD)*, 2017.
- [5] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *Proc. of Int. Conf. on World Wide Web*, 1998.
- [6] S. Bulathwela, S. Kreitmayer, and M. Pérez-Ortiz. What's in it for me? augmenting recommended learning resources with navigable annotations. In *Proceedings of the 25th International Conference on Intelligent User Interfaces Companion*, IUI '20, 2020.
- [7] S. Bulathwela, M. Perez-Ortiz, A. Lipani, E. Yilmaz, and J. Shawe-Taylor. Predicting engagement in video lectures. In *Proc. of Int. Conf. on Educational Data Mining*, EDM '20, 2020.
- [8] S. Bulathwela, M. Perez-Ortiz, E. Novak, E. Yilmaz, and J. Shawe-Taylor. Peek: A large dataset of learner engagement with educational videos, 2021.
- [9] S. Bulathwela, M. Perez-Ortiz, E. Yilmaz, and J. Shawe-Taylor. Truelearn: A family of bayesian algorithms to match lifelong learners to open educational resources. In *AAAI Conference on Artificial Intelligence*, AAAI '20, 2020.
- [10] S. Bulathwela, M. Perez-Ortiz, E. Yilmaz, and J. Shawe-Taylor. Semantic truelearn: Using semantic knowledge graphs in recommendation systems. In *Proc. of First Int. Workshop on Joint Use of Probabilistic Graphical Models and Ontology*, PGMonto '21, 2021.
- [11] S. Bulathwela, E. Yilmaz, and J. Shawe-Taylor. Towards Automatic, Scalable Quality Assurance in Open Education. In *Workshop on AI and the United Nations SDGs at Int. Joint Conf. on Artificial Intelligence*, 2019.
- [12] R. Burke. Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction*, 12(4):331–370, 2002.
- [13] Y. Choi, Y. Lee, D. Shin, J. Cho, S. Park, S. Lee, J. Baek, C. Bae, B. Kim, and J. Heo. Ednet: A large-scale hierarchical dataset in education. In *International Conference on Artificial Intelligence in Education*, pages 69–73. Springer, 2020.
- [14] K. Clements and J. Pawlowski. User-oriented quality for oer: understanding teachers' views on re-use, quality, and trust. *Journal of Computer Assisted Learning*, 28(1), 2012.
- [15] K. Collins-Thompson, P. N. Bennett, R. W. White, S. de la Chica, and D. Sontag. Personalizing web search results by reading level. In *Proc. of Conf. on Information and Knowledge Management*, 2011.
- [16] P. Covington, J. Adams, and E. Sargin. Deep neural networks for youtube recommendations. In *Proc. of ACM Conf. on Recommender Systems*, 2016.
- [17] N. Craswell, D. Campos, B. Mitra, E. Yilmaz, and B. Billerbeck. Orcas: 20 million clicked query-document pairs for analyzing search. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management*, CIKM '20, page 2983–2989, New York, NY, USA, 2020. Association for Computing Machinery.
- [18] D. H. Dalip, M. A. Gonçalves, M. Cristo, and P. Calado. A general multiview framework for assessing the quality of collaboratively created content on web 2.0. *Journal of the Association for Information Science and Technology*, 2017.
- [19] D. H. Dalip, M. A. Gonçalves, M. Cristo, and P. Calado. Automatic assessment of document quality in web collaborative digital libraries. *Journal of Data and Information Quality*, 2(3), Dec. 2011.
- [20] D. Davis, G. Chen, C. Hauff, and G. Houben. Activating learning at scale: A review of innovations in online learning strategies. *Comput. Educ.*, 125:327–344, 2018.
- [21] D. Davis, D. Seaton, C. Hauff, and G. Houben. Toward large-scale learning design: categorizing course designs in service of supporting learning outcomes. In *Proceedings of the Fifth Annual ACM Conference on Learning at Scale, London, UK, June 26-28, 2018*, pages 4:1–4:10, 2018.
- [22] M. A. Dewan, M. Murshed, and F. Lin. Engagement detection in online learning: a review. *Smart Learning Environments*, 6(1):1, 2019.
- [23] P. J. Guo, J. Kim, and R. Rubin. How video production affects student engagement: An empirical study of mooc videos. In *Proc. of the First ACM Conf. on Learning @ Scale*, 2014.
- [24] M. Horta Ribeiro and R. West. Youiverse: Large-scale channel and video metadata from english-speaking youtube. *Proceedings of the International AAAI Conference on Web and Social Media*, 15(1), 2021.
- [25] A. Kaur, A. Mustafa, L. Mehta, and A. Dhall. Prediction and localization of student engagement in the wild. In *2018 Digital Image Computing: Techniques and Applications (DICTA)*, pages 1–8. IEEE, 2018.
- [26] S. Kim, W. Kim, Y. Jang, S. Choi, H. Jung, and H. Kim. Student knowledge prediction for teacher-student interaction. *Proceedings of the AAAI Conference on Artificial Intelligence*,

- 35(17):15560–15568, 2021.
- [27] M. Kurdi, N. Albadi, and S. Mishra. “think before you upload”: an in-depth analysis of unavailable videos on youtube. *Social Network Analysis and Mining*, 11(1):1–21, 2021.
- [28] A. S. Lan, C. G. Brinton, T.-Y. Yang, and M. Chiang. Behavior-based latent variable model for learner engagement. In *Proc. of Int. Conf. on Educational Data Mining*, 2017.
- [29] Z. MacHardy and Z. A. Pardos. Evaluating the relevance of educational videos using bkt and big data. In *Proc. of Int. Conf. on Educational Data Mining*, 2015.
- [30] M. Mendicino, L. Razzaq, and N. T. Heffernan. A comparison of traditional homework to computer-supported homework. *Journal of Research on Technology in Education*, 41(3):331–359, 2009.
- [31] A. Ntoulas, M. Najork, M. Manasse, and D. Fetterly. Detecting spam web pages through content analysis. In *Proc. of Int. Conf. on World Wide Web*, 2006.
- [32] M. Perez-Ortiz, C. Dormann, Y. Rogers, S. Bulathwela, S. Kreitmayer, E. Yilmaz, R. Noss, and J. Shawe-Taylor. X5learn: A personalised learning companion at the intersection of ai and hci. In *26th International Conference on Intelligent User Interfaces - Companion*, IUI ’21 Companion. Association for Computing Machinery, 2021.
- [33] M. Perez-Ortiz, A. Mikhailiuk, E. Zerman, V. Hulusic, G. Valenzise, and R. K. Mantiuk. From pairwise comparisons and rating to a unified quality scale. *IEEE Transactions on Image Processing*, 29:1139–1151, 2019.
- [34] A. Ramesh, D. Goldwasser, B. Huang, H. Daume III, and L. Getoor. Learning latent engagement patterns of students in online courses. In *Proc. of AAAI Conference on Artificial Intelligence*, 2014.
- [35] J. Shi, C. Otto, A. Hoppe, P. Holtz, and R. Ewerth. Investigating correlations of automatically extracted multimodal features and lecture video quality. In *Proceedings of the 1st International Workshop on Search as Learning with Multimedia Information*, SALMM ’19, page 11–19, New York, NY, USA, 2019. Association for Computing Machinery.
- [36] R. Syed and K. Collins-Thompson. Optimizing search results for human learning goals. *Inf. Retr. J.*, 20(5):506–523, 2017.
- [37] K. Taghipour and H. T. Ng. A neural approach to automated essay scoring. In *Proc. of Conf. on Empirical Methods in Natural Language Processing*, 2016.
- [38] UNESCO. Open educational resources (oer). <https://en.unesco.org/themes/building-knowledge-societies/oer>, 2021. Accessed: 2021-04-01.
- [39] M. Warncke-Wang, D. Cosley, and J. Riedl. Tell me more: An actionable quality model for wikipedia. In *Proc. of Int. Symposium on Open Collaboration*, 2013.
- [40] S. Wu, M. Rizoiiu, and L. Xie. Beyond views: Measuring and predicting engagement in online videos. In *Proc. of the Twelfth Int. Conf. on Web and Social Media*, 2018.
- [41] F. Xu, L. Wu, K. P. Thai, C. Hsu, W. Wang, and R. Tong. MUTLA: A large-scale dataset for multimodal teaching and learning analytics. *CoRR*, abs/1910.06078, 2019.
- [42] J. Yu, G. Luo, T. Xiao, Q. Zhong, Y. Wang, W. Feng, J. Luo, C. Wang, L. Hou, J. Li, et al. Mooccube: a large-scale data repository for nlp applications in moocs. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3135–3142, 2020.

Is there Method in Your Mistakes? Capturing Error Contexts by Graph Mining for Targeted Feedback

Maximilian Jahnke
Ostfalia University of Applied Sciences
maxi.jahnke@ostfalia.de

Frank Höppner
Ostfalia University of Applied Sciences
f.hoeppner@ostfalia.de

ABSTRACT

The value of an instructor is that she exactly recognizes what the learner is struggling with and provides constructive feedback straight to the point. This work aims at a step towards this type of feedback in the context of an introductory programming course, where students perform program execution tracing to align their understanding of Java instructions with reality. The students' submissions are analyzed for repeating mistakes across different exercises by representing the context surrounding the error by a graph and applying graph mining techniques to discover their common grounds. The patterns need to be annotated only once and help to address misconceptions of individual students. They may also be used to select follow-up exercises automatically, that contain the same intricacy.

Keywords

graph mining, formative feedback, teaching programming, misconception

1. INTRODUCTION

Automatized grading is integrated (to varying extent) in many educational systems. It is often limited to comparing a correct answer against the submission (or applying unit tests), but is nevertheless welcomed by both, students and instructors, for different reasons: While students appreciate immediate feedback, instructors are relieved that the burden of manual grading is lifted. But a student who has missed the point of some problem and thus repeatedly gets the answers wrong would benefit much more from personal feedback from an instructor who is capable of recognizing and directly addressing the student's problem. And if none of the submissions is inspected manually any more, the instructor withholds his diagnostic skills that may otherwise have been proven useful to identify misconceptions of several course members. In this work we investigate – for a particular type of exercises in programming courses – how this situation may be improved.

M. Jahnke and F. Höppner. Is there method in your mistakes? Capturing error contexts by graph mining for targeted feedback. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 422–429, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853193>

We consider an introductory programming course in Java for students who have no programming experience yet. While students may struggle with programming for a large number of reasons (e.g. fighting with the development environment, getting a grip on computational thinking, etc.), in this work we concentrate on a single aspect only: instruction comprehension. Especially for students who have no experience in programming, we observe in lab discussions that their perception of an instruction often deviates from reality. Sticking to a wrong mental model of program execution makes it difficult to write correct programs. A human advisor can (1) discover the misconception (even if it did not occur before), (2) explain the problem (and link it to the course material), and (3) challenge the student afterwards (to monitor progress). The research question of this paper is how such an ideal instructor might be mimicked (in the given context). In particular, is it possible to support the instructor in the discovery of new misconceptions without falling back to time-consuming manual inspection of all submissions?

2. RELATED WORK

The importance of dedicated feedback has been acknowledged by many researchers ([5] gives a review). In the context of programming, several approaches derive precise commands to fix the mistakes of a submitted solution (e.g. [9, 11]). While being helpful to fix a technical problem, applying detailed instructions mechanically will unlikely trigger a change of the students' mental model – the same mistake might be made again if nothing can be learned from the feedback (e.g. because it is lacking explanations). Different hint levels are offered as a solution in [10]: On the first level, a hint points to the problem but still requires the *right student action*, while a hint on the second level enables them to fix the problem mechanically. When misconceptions are known beforehand, their discovery in a submission turns into a classification problem. Recently a number of deep learning approaches have been proposed to predict whether a student will complete an exercise [14], the code exhibits certain logic errors [6], or some free-text answer hints towards a common misconception [7]. For all those misconceptions a dedicated feedback may be prepared only once (manually by the instructor) and whenever it is predicted in a new submission the feedback can be provided automatically. This is effective for *known misconceptions*, but such approaches do not help to discover *new misconceptions*.

One of the few approaches that may also help to discover *new problems* (and is therefore close to what we have in

mind) is suggested in [8], where the focus is on programming bugs. Common coding errors (for the same exercise) are discovered by finding subtrees (in the submissions' abstract syntax trees¹ (AST)) that correlate with failed unit tests, hinting at *typically wrong* code fragments for the given exercise. As with other approaches, an instructor provides feedback for a buggy subtree that will be delivered later whenever a similar subtree occurs again. The focus of these explanations is, however, restricted to the exercise at hand – a buggy subtree in one exercise may be correct in another.

3. PROGRAM EXECUTION TRACES

When an instructor explains some concepts in class it is difficult to verify that the students perceive the concepts in the intended way. This holds in general and for instructions of a programming language in particular. If a student has only vague ideas about stack and heap, call-by-value/reference, array organization, etc., programming errors will happen sooner or later. But since code will be executed by *man-made* machines, we know the ground truth (just find a debugger) and can ask students to simulate program execution (to some extent). Paper-based analysis of variable tracing exercises have been used to investigate misconceptions in the literature [1, 2, 4, 12]. Gaining an understanding of the mechanics how the language implementation worked, was also identified as a key issue in [13]. Employing tracing exercises in class has been reported to increase the students' performance in [3]. We thus consider execution traces as a helpful tool to align mental models and encourage students to verify their understanding by filling out traces: for every executed code line the content of all variables has to be documented.

Fig. 1 shows a Java code snippet on the right that uses a mixture of language elements for demonstration purposes. We ask the students to fill out an execution trace as shown on the left, which is basically a large table where each line corresponds to a complete memory snapshot – ordered in time from top to bottom. For each row, the student has to provide the line number that is executed next (e.g. execution starts with the first line 11 of the main function). The state of all variables must be entered for each row after the corresponding code line has been executed. For instance, when line 12 got executed there is a new variable `x` on the stack which points to the heap address `0x0`, where an array of length 2 has been instantiated and initialized with `null` references. Students enter such traces in a web application, which marks erroneous lines but not the exact error position to make the students think about their input and make *trial and error* strategies less attractive. The downside of this practice is that students may get stuck at some point because they are simply not aware of what they are doing wrong. A lab advisor, pointing them to the nature of their problem, would be appreciated by them.

4. OUTLINE OF THE APPROACH

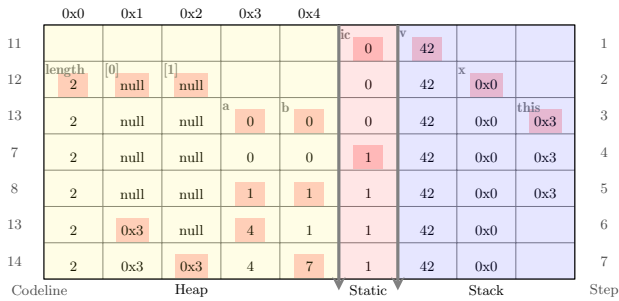
Although a full trace may consist of a considerable number of inputs (cf. Fig. 1), most of the student's input is not very informative because it (hopefully) corresponds to the already known solution. The valuable sources of information are the deviations alone, because they may hint to miscon-

ceptions. A wrong value in the trace alone does not tell us anything about possible reasons *why* this error was made. This can only be judged if more context is given, that is, the situation in which the error occurred. If errors re-occur in *similar situations*, the likelihood of a misconception rises – otherwise it just might be a typo. The *context* is the Java program itself. However, similar situations do not necessarily correspond to identical source lines. For the sake of simplicity, let us consider the case of a student who is confused by assignments where the same variable occurs on the left and the right hand side of the assignment. (Sometimes beginners read instructions with `=` as a mathematical equation rather than an assignment.) Fig. 2 shows a simplistic code example together with the correct trace. Can we tell from the student's input if he is affected by this misconception? There are two places in the trace (marked in yellow) which we can associate with this misconception: the value of variable `b` after executing line 7 and the value of variable `a` after executing line 9 as both lines exhibit the discussed kind of assignment. If both places – or *similar* places in other traces – are wrong, chances rise that assignments have not yet been understood correctly. We have identified these two places in the trace because we *knew* what kind of misconception we were looking for. At these places the probability $P(\text{error}|\text{misconception})$ of an faulty input rises, given the misconception holds. As it may be difficult to come up with an operational definition of *misconception*, we retreat to $P(\text{error}|\text{context})$ and will define *context* as a situation or pattern that can be matched against the traces and the source code.

Recalling the research question, we seek a mechanism that identifies both, the context (of a potential misconception) as well as the positions affected by it. If the context is easily understandable, such a mechanism would enable us to mimic the behaviour of a valuable instructor with only little manual intervention: (1) discovery: if errors in an (automatically derived) context accumulate substantially (across all students), it is likely the context captures a problematic situation. The instructor inspects the context once and decides what a likely root cause might be. Once it got accepted as a misconception in this way, it can be discovered in subsequent submissions automatically. If the errors accumulate for some individual student, we conclude that he suffers from this misconception. (2) feedback: The instructor writes a short explanation when the problem was discovered for the first time (or may link it to existing material). As with other approaches, this feedback can be delivered in subsequent occurrences. (3) challenge: Beyond the textual feedback, we need a challenge to check for an improved understanding. We characterize a student by a set of *pending misconceptions* and, in the same way, we characterize exercises by the contexts they contain. A suitable challenge is thus a challenge that matches the students profile best (in the fashion of an adaptive recommendation system).

A *context* must address properties of the source code **and** the trace: We want to align errors in the trace to properties of the source code. The plain code (Fig. 2(right)) is therefore transformed into a graph as shown in Fig. 3 as follows: The first step is the creation of an abstract syntax tree, which already provides most of the nodes in the graph. Starting from the top node we can see the method declaration (of

¹An AST is build by a parser and represents the syntactical structure of code while omitting some less relevant details.



```

1 package test;
2 public class Demo {
3   static int ic;
4   int a,b;
5   public Demo() {
6     ic++;
7     a=ic; b=ic*ic;
8   }
9   public static void main(String[] args) {
10    int v = 42;
11    Demo[] x = new Demo[2];
12    x[0] = new Demo(); x[0].a=4;
13    x[1] = x[0]; x[1].b=7;
14 } }

```

Figure 1: For the example code on the right, the corresponding execution trace is shown on the left. The different memory areas stack, heap, and static data are shown in blue, yellow, and red, resp.

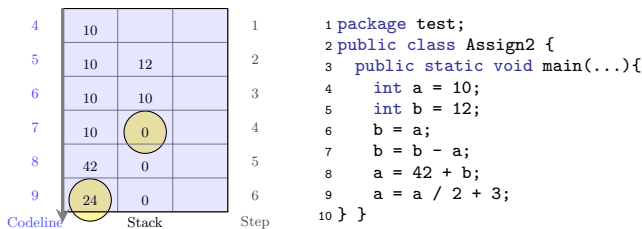


Figure 2: Code (right) and corresponding trace (left). Yellow marks indicate where the misconception may be observed.

main), which contains a body (instruction block `{}`) with 6 statement nodes, two declarations and four assignments. But the AST does not yet suit the trace, it represents the static structure of the code, but the trace is concerned with the dynamic evolution of the variables during code execution. We bridge this gap by two changes:

1. Every occurrence of the *same variable* in the code introduces a new node in the AST, but it has a unique position in memory (and thus a unique position in the trace). Therefore all AST nodes that refer to the same variable are united to a single node, turning the tree into a graph. In Fig. 3 both nodes for the variables `a` and `b` are shaded in gray. Every incoming edge into one of these variable nodes correspond to the use of the variable in the code.
2. The content of the trace is aligned with the code lines, but by looking at an AST graph one cannot tell which instruction comes from which code line. So we introduce a new node for each code line (shown in blue in Fig. 3) and link them with all instructions from this code line. (For instance, the rightmost assignment node in Fig. 3 occurs in line 9 of Fig. 2.) We capture the program flow by connecting code line nodes that may get executed in sequence. In the example we have no loops or conditional statements, so the nodes connect linearly (*Line 5 to Line 6, Line 6 to 7, etc.*)

This graph combines information from code and trace, but does not yet encode where the student entered a wrong value. The error position in the trace is identified by row

and column (cf. yellow circles in Fig. 2): The column corresponds to the variable and the row corresponds to a time step during execution, which always refers to a specific code line. An additional *focus node* (orange in Fig. 3) thus encodes the error position by connecting to both, the variable for which an erroneous value has been entered in the trace², and the code line node whose execution has been traced.

We denote such a graph G as a **code graph**. A *context* may now be represented as a subgraph C of G . In Fig. 3 such a subgraph has been highlighted with red lines. We read the subgraph as follows: We focus on a code line (here: line 7) that contains an assignment where the variable (here: `b`) on the left hand side also occurs in the expression on the right hand side of the assignment. The context graph is also shown isolated in Fig. 4(left). If this context accumulates a high number of student errors, it can be shown to the instructor and should be reasonably simple to interpret. The instructor may then decide that this context represents a misconception and writes a short explanation for the context pattern as indicated in Fig. 4(right). The text template may refer directly to nodes in the context graph (e.g. `[varname_2]`). These references are replaced with the true line numbers or variable names from the real exercise the student is currently working on. This links the feedback text very tight to the exercise just submitted by the student.

To match such a subgraph to other occurrences, it has to be *anonymized*, for instance, the variable names have to be replaced by placeholders 'varname' (as we have already seen in Fig. 4). Variables have unique nodes in the graph and we no longer need the variable's name for disambiguation. Other graph transformations may be applied to compensate structural differences in the AST that are not helpful for our purposes. For instance, we want the context to match twice in the example code of Fig. 2, but the subtree for the expression on the right hand side of `b=b-a` is less complex than that of `a=a/2+3`, so the code graph is structurally different. To ensure that a context can still match both occurrences, we apply two measures: Firstly, the graph is transformed to simplify some technical details (we collapse a tree of expression nodes into a single expression node). Secondly, we extend the expressiveness of a context: Rather than only simple edges we allow, e.g., for transitive connec-

²only if there is one; line number errors do not have an associated variable

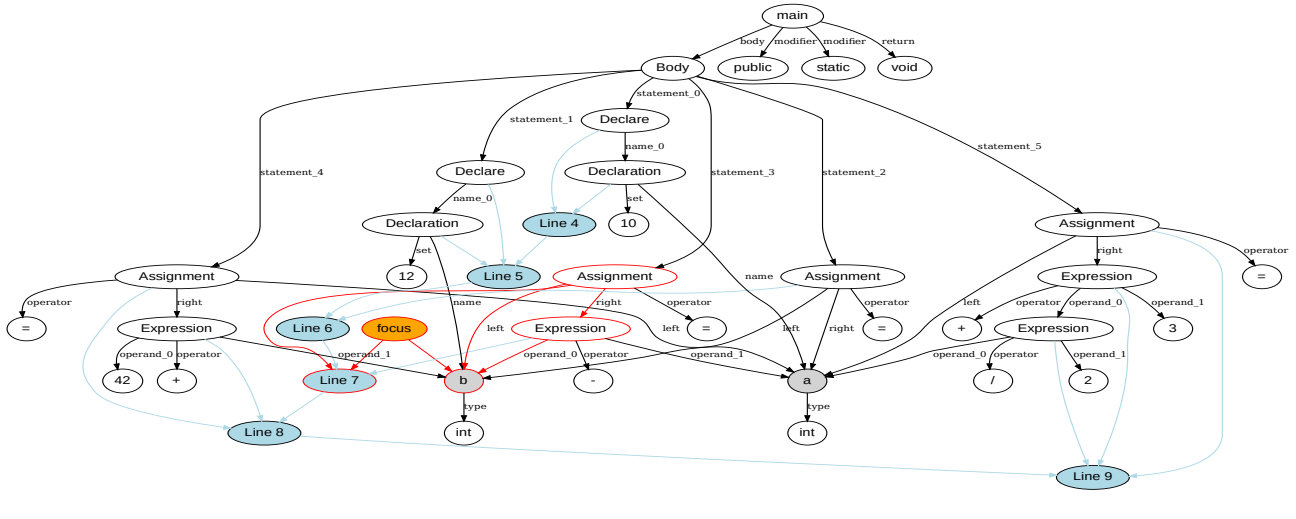


Figure 3: A graph that fuses information from the abstract syntax tree (black), the execution trace (blue) and a potential error position in the trace (orange). A potential misconception can be characterized as a subgraph (red).

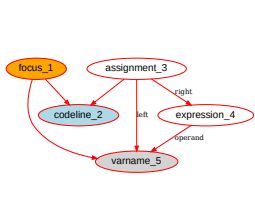


Figure 4: Context graph and excerpt from a text template to deliver meaningful feedback to a student with a potential misconception.

tions (a path of arbitrary length from the expression node to the variable node instead of a direct neighbour).

5. ERROR CONTEXT DISCOVERY

In this section we briefly sketch a graph mining approach to discover context patterns automatically from data.

5.1 Code and Context Graphs

Our input consists of various Java sources, the correct solution trace and the student’s input. The example graph shown in Fig. 3 *focuses* on a specific position in the trace (row/column as encoded by the `codeline` and `varname` nodes linked to the focus node). We create corresponding graphs for every position (row/column) in a trace. At first glance, this affects only the edges of the focus node while the rest of the graph remains unaltered (for a given tracing exercise). However, the relevant context for entering a wrong value in the trace can be narrowed down to the *already passed code lines*: The reason for a mistake cannot be found in source code lines yet to come. Knowing which code line we are focussing at, we remove all AST nodes that correspond to code beyond the focus line.

Apart from the transformations already mentioned in Sect. 4 (renaming variable nodes, flattening expressions), we intro-

duce extra edges from an instruction block to the very first and last statement of the block (labelled `first` and `last`). Other transformations replace numbers by a `literal` label, or re-insert information about variable names: whenever two variables have the same name (before replacing them) we add a `samenname`-edge between them. A code graph is then defined as follows: A triple $G = (V, E, \lambda)$ is called **code graph**, iff V is a non-empty set of nodes, $E \subseteq V \times V$ is a set of directed edges between nodes, $\lambda : (V \cup V \times V) \rightarrow L$ is a function that assigns labels (from a set of labels L) to nodes and edges, and there is exactly one $v \in V$ with $\lambda(v) = \text{focus} \in L$.

We assume that a set \mathcal{G} of all code graphs is given. Having a graph $G \in \mathcal{G}$ for every individual cell (of every trace), we associate counters with them to track how often the respective cell has been entered correctly or not in the submissions. We denote the (absolute) frequencies of correct and incorrect entries by $\text{pos}(G)$ and $\text{neg}(G)$, resp. For a given context C the core operation is to decide whether C can be found in a given code graph G , which is abbreviated as $C \sqsubseteq G$. Then we define

$$P(\text{error}|\text{context } C) = \frac{\sum_{G \in \mathcal{G}, C \sqsubseteq G} \text{neg}(G)}{\sum_{G \in \mathcal{G}, C \sqsubseteq G} \text{pos}(G) + \text{neg}(G)} \quad (1)$$

The graph mining algorithm then searches for a context C that maximizes $P(\text{error}|\text{context } C)$. Formally, we define a context pattern as follows: A tuple $P = (V, E, \lambda, \tau)$ is called **context graph**, iff V is a non-empty set of nodes, $E \subseteq V \times V$ is a set of directed edges between nodes, $\lambda : (V \cup V \times V) \rightarrow L \cup \{*\}$ assigns labels to nodes and edges (where $*$ is a wildcard label), there is exactly one $v \in V$ with $\lambda(v) = \text{focus}$, and $\tau : E \rightarrow \{N, X, T, W\}$ classifies edges into one of four types: normal (N), excluding (X), transitive (T) or wildcard (W).

The semantics of the edge types and the wildcard label (which matches any other label) is defined along with the definition of graph inclusion \sqsubseteq as follows: A context graph $C = (V_C, E_C, \lambda_C, \tau_C)$ is contained in a code graph $G =$

(V_G, E_G, λ_G) , $\mathbf{C} \sqsubseteq \mathbf{G}$ for short, iff there is a bijective function $\sigma : V_C \rightarrow V_G$ such that all node labels match ($\forall v \in V_C : \lambda_C(v) = \lambda_G(\sigma(v))$) or the context label is a wildcard ($\lambda_C(v) = *$), and for all edges in C there are *corresponding edges* in G . An edge $(u, v) \in E_C$ has a corresponding edge if: (1) for normal edges ($\tau((u, v)) = N$) we have $(\sigma(u), \sigma(v)) \in E_G \wedge \lambda_C((u, v)) = \lambda_G((\sigma(u), \sigma(v)))$, (2) for transitive edges ($\tau((u, v)) = T$) there is a path (w_1, w_2, \dots, w_k) in G with $\sigma(u) = w_1$, $\sigma(v) = w_k$, and $\forall i : \lambda_C((u, v)) = \lambda_G((w_{i-1}, w_i))$, (3) for wildcard edges ($\tau((u, v)) = W$) we have $(\sigma(u), \sigma(v)) \in E_G$, (4) for excluding edges ($\tau((u, v)) = X$) we have $(\sigma(u), \sigma(v)) \in E_G \wedge \lambda_C((u, v)) \neq \lambda_G((\sigma(u), \sigma(v)))$.

5.2 Mining Algorithm

In contrast to other graph mining algorithms, our notion of subgraph inclusion (\sqsubseteq) is more complicated due to the different edge types. But we benefit from the fact that we have a clear starting point for inclusion tests because a unique *focus node* is required in both, code graphs and context graphs. The out-degree for many nodes is limited by the fixed syntax of Java instructions, but this does not hold for nodes like instruction blocks or expressions.

From a context graph that may serve as an indicator for a misconception we expect a higher rate of faulty student inputs than on average. A context graph C matches various cells in the traces, but the total number of correct inputs is usually much larger than the total number of incorrect answers. To account for this imbalance, we normalize the correct (and incorrect) number of answers in context C :

$$p_C = \frac{\sum_{G \in \mathcal{G}, C \sqsubseteq G} \text{pos}(G)}{\sum_{G \in \mathcal{G}} \text{pos}(G)}, \quad n_C = \frac{\sum_{G \in \mathcal{G}, C \sqsubseteq G} \text{neg}(G)}{\sum_{G \in \mathcal{G}} \text{neg}(G)}$$

We define an objective function f to rate context C as:

$$f(C) \stackrel{\text{max!}}{=} \log_2 \left(\frac{n_C}{p_C} \right) \quad (2)$$

If the fractions n_C and p_C are about the same, the answers in context C do not distinguish from the average rate over all cells and we obtain $f(x) = \log_2(1) = 0$. If we manage to find a context C that covers twice as many erroneous cases (than on average) and half as many correct cases (than on average), it evaluates to $\log_2 \left(\frac{2}{\frac{1}{2}} \right) = 2$. We perform a beam search in the space of context graphs to identify the context that maximizes (2). Details about the search algorithm are given in the appendix A.

6. EXPERIMENTAL EVALUATION

When the students filled out the execution traces, the submitted solution may contain multiple errors. It is likely that the first error in the trace causes subsequent faults: If the student enters a wrong value in some cell, but this variable is not altered in subsequent code lines, the wrong value remains and invalidates multiple trace lines. Therefore we consider only the first error from each submission, which most likely contains its root cause. In total, there were roughly 100 accounts (some were used only a few times) and all traces together consisted of $|\mathcal{G}| = 7128$ different input cells. All students together pressed the ‘evaluation’ button almost 92000 times. The total number of correct and incorrect entries in the submissions was 2.4M and 64k, resp.

6.1 Discovered Contexts

In this section we present and discuss a few of the context graphs that were automatically discovered by the algorithm from Sect. 5. The objective is to investigate meaningfulness and interpretability of the discovered contexts. In the figures, the node labels carry unique numbers for easier reference (e.g. `focus_0` denotes node #0) and the edges have a suffix indicating their edge type (e.g. N for normal edge) and a varying line style (N : solid, W : dotted, X : red).

The context in Fig. 5 addresses the lifetime of local variables. Many students have a superfluous entry on the stack in this context. The variable #2 has been declared #3 in an instruction block #6, whose last statement is an assignment (#5, connects to #6 with edge labelled `last`). This is the last statement of the block #6 and the superfluous variable has been observed in the next executed line (#1, connected to #4 with an edge labelled `pre vex` (previously executed)). This code line is thus the first after the end of block #6 and variable #2 should have vanished because its lifetime ended.

A somewhat surprising context is shown in Fig. 6, which describes a code line #1 with an `if`-statement #5, where the focussed variable #2 is somehow involved in the control-expression #6 of the `if`-statement. An inspection of the submitted traces reveals that some students stored the boolean result of the control expression on the stack. (Probably the students thought that the evaluation result must be traced *somehow*, but as it does not affect any variable it is not.)³

Fig. 7 captures a context where local variables and attributes are mixed up. The focus is on a local variable #2 (local because of its path to #6) in a function #6 of a class #7 that also contains a field declaration #8. The local variable has a `same name`-edge to another variable, which causes the confusion. Although the equally named variable #4 is not connected to the field declaration #8, it corresponds to it in all matching code graphs. So the student was not aware which variable was addressed in the code and picked the wrong one.

Some contexts may not really relate to code understanding, but point at pitfalls when starting to use the web application. Programs usually start with some variable declaration, so the attention is initially on the stack – and filling out the line number is easily forgotten: In Fig. 8 the expected code line (whose line number was not entered correctly) contains a declaration #2, which is the first instruction in the body #3 of a function #8 – it thus most likely corresponds to the very first code line that has to be traced.

Discussion: The graph mining approach delivered various meaningful context patterns that correspond well to our expectations as well as some unexpected patterns. Once an instructor gets used to them, the graphs are comparatively easy to interpret, which was a requirement for the envisaged approach to solve the research question in Sect. 4 – although sometimes it may be necessary to have a look at the submissions to get an idea what types of error were made. Not all of the discovered patterns were useful, some are too general in nature or mix up several problems. But compared to the

³We consider it unlikely that they tried to mimic the JVM-internal evaluation of expressions.

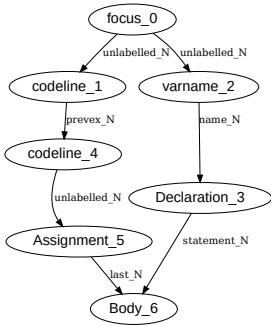


Figure 5: Error context for the lifetime of variables.

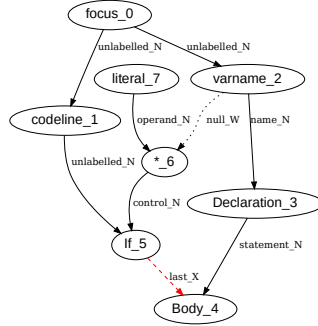


Figure 6: Effect of evaluating expressions.

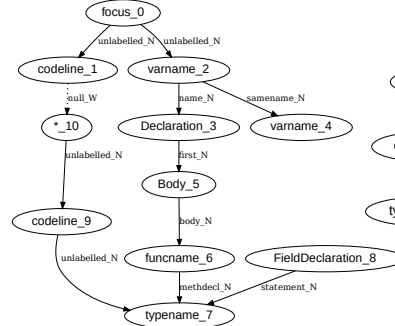


Figure 7: Confusing equally named variable and attribute.

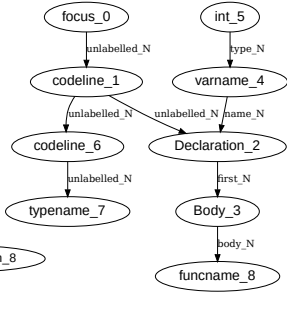


Figure 8: Line number is missing at start of an exercise.

Table 1: Based on the performance over a series of exercises, the students are classified in three groups S_+ , S_\uparrow , S_- . The table shows how many students belong to which group (percentage) and the median number of exercises per group.

context	group size %			practise #		
	S_\uparrow	S_+	S_-	S_\uparrow	S_+	S_-
line number	39.6	27.9	32.4	52	26	25
missing stack	29.8	51.8	18.2	26	19	19
wrong heap value	18.8	64.1	16.9	48	27	20
expected reference	50.2	36.1	13.6	44	31	23
Fig. 5	33.3	44.4	22.2	16	12	13
Fig. 7	20.0	40.0	40.0	23	16	19

time spent on inspecting many raw submissions manually, it takes an instructor much less time to select useful patterns and write an accompanying explanation.

6.2 Evaluating Demand and Progress

Whenever a new concept or instruction has been introduced during the lecture, one group of students (S_+) will have no problems at all with the new instruction type and will enter the traces flawlessly. The second group, which experiences problems, splits up further: the group of reflective students (S_\uparrow) will notice the knowledge gap, may read the course material or offered help text again, and will at some point close the gap – in the future this type of mistake will happen only occasionally and incidentally. We expect a significant reduction in the error rate for this group of students. The remaining group (S_-) does, for some reason, not improve. We can distinguish these three groups on the level of individual misconceptions as follows: For a given error context, we collect all submissions of a student and order them in time. For any given point in time t we may characterize the students performance *before* and *after* t by counting how often the student got the trace in this context right or not. If we can find a point in time t such that there is a *significant improvement* in the distribution of mistakes before and after t , we say that this student belongs to group S_\uparrow . We assign students with a rather low error rate (say, consistently $\leq 10\%$) to group S_+ (at such a low error rate it is very difficult to improve significantly). The remainder belongs to group S_- .

Table 1 shows the size of the groups (percentage) and the median number of exercises per group. The rows aggregate a few different context graphs related to line number errors, missing values on the stack, wrong values on the heap, and situations in which a reference was expected but a number was entered. The numbers for the context patterns of Fig. 5 and 7 are also shown. The table shows that members of S_\uparrow have traced more exercises than any other group (sometimes more than twice as many as group S_-), so the success is correlated with the gained practice. It is not surprising that the median number of practised exercises per context in group S_+ is more similar to group S_- than to S_\uparrow : If they do it right from the beginning, there is less need for further practising.

The experiments show that we can measure (by means of the statistical test) whether an individual student performs significantly worse on a context pattern than the course on average and whether he has improved over time. Similar to feedback from a lab advisor, which is rare but to the point, we may provide feedback to detect misconceptions only if the students make mistakes in the error context significantly more often than the course average (rather than annoying a student with hints at the first incidental mistake). The numbers from Tab. 1 also show that there is demand for support, a relevant fraction of the course did not manage to significantly improve on their own.

7. CONCLUSION

A considerable fraction of the participants did not manage to overcome tracing difficulties on their own, which shows the demand for constructive feedback that goes beyond a right/wrong classification. The automatically discovered context graphs can be assessed by an instructor with comparatively low effort, because they are quite easy to interpret and meaningful. This paves the way for identifying even new misconceptions in reasonable time. The context graphs allow us to analyse the students' state of knowledge and consequently deliver purposeful feedback when needed. It also enables us to recommend suitable exercises to check an improved understanding afterwards. Just like a human advisor, we may congratulate the student if the error rate decreased significantly. This will bring us, in response to the research question, quite close to a human advisor.

8. REFERENCES

- [1] P. Baffes and R. Mooney. Refinement-based student modeling and automated bug library construction. *Journal of Artificial Intelligence in Education*, 7(1):75–116, 1996.
- [2] R. Bornat, S. Dehnadi, and Simon. Mental models, consistency and programming aptitude. *Conferences in Research and Practice in Information Technology Series*, 78(1986):53–61, 2008.
- [3] M. Hertz and M. Jump. Trace-Based Teaching in Early Programming Courses. *Proc. Techn. Symp. Computer Science Education*, pages 561–566, 2013.
- [4] A. N. Kumar. Explanation of step-by-step execution as feedback for problems on program analysis, and its generation in model-based problem-solving tutors. *Technology, Instruction, Cognition and Learning*, 4(1):65–107, 2006.
- [5] N.-T. Le. A classification of adaptive feedback in educational systems for programming. *Systems*, 4(2):22, 2016.
- [6] D. Miao, Y. Dong, and X. Lu. PIPE: predicting logical programming errors in programming exercises. In *Proc. Int. Conf. Educational Data Mining*, 2020.
- [7] J. J. Michalenko, A. S. Lan, A. E. Waters, P. Grimaldi, and R. G. Baraniuk. Data-mining textual responses to uncover misconception patterns. In *Proc. Int. Conf. Educational Data Mining*, 2017.
- [8] A. Nguyen, C. Piech, J. Huang, and L. Guibas. Codewebs: scalable homework search for massive open online programming courses. In *Proc. Int. Conf. on World Wide Web*, pages 491–502, 2014.
- [9] T. W. Price, R. Zhi, and T. Barnes. Evaluation of a data-driven feedback algorithm for open-ended programming. In *Proc. Int. Conf. Educational Data Mining*, 2017.
- [10] K. Rivers and K. R. Koedinger. Data-driven hint generation in vast solution spaces: a self-improving python programming tutor. *Int. J. Artif. Intell. Educ.*, 27(1):37–64, 2017.
- [11] R. Singh, S. Gulwani, and A. Solar-Lezama. Automated feedback generation for introductory programming assignments. In *Proc. Conf. Programming Language Design and Implementation*, pages 15–26, 2013.
- [12] K. Stephens-Martinez, A. Ju, K. Parashar, R. Ongowarsito, N. Jain, S. Vekat, and A. Fox. Taking advantage of scale by analyzing frequent constructed-response, code tracing wrong answers. In *Proceedings of the 2017 ACM Conference on International Computing Education Research*, pages 56–64. ACM, 2017.
- [13] M. W. van Someren. What’s wrong? understanding beginners’ problems with prolog. *Instructional Science*, 19(4-5):257–282, 1990.
- [14] L. Wang, A. Sy, L. Liu, and C. Piech. Learning to represent student knowledge on programming exercises using deep learning. In *Proc. Int. Conf. Educational Data Mining*, 2017.
- [15] B. Weisfeiler and A. Lehman. A reduction of a graph to a canonical form and an algebra arising during this reduction. *Nauchno-Technicheskaya Informatsia*, 2(9):12–16, 1968.

APPENDIX

A. MINING ALGORITHM

We incrementally explore the space of all possible context graphs by extending it step by step (add an edge and/or node). To evaluate which extension improves the objective function most, we need to match a new context C' to the graphs $G \in \mathcal{G}$, that is, we have to make sure that a bijective mapping σ exists. Finding the possible mappings σ each time from scratch would imply substantial computational cost (especially as we have matched the first $n - 1$ of the n nodes in the previous expansion already), so we keep track of the mappings σ that have been used for the current context so far. The bookkeeping is illustrated in Fig. 9. On the top left a context graph and on the top right excerpts from a code graph are shown. The node sets V consist of natural numbers, which are shown in subscript next to the assigned labels. Initially we start with a context graph C_0 that consists of a **focus** node alone ($V = \{1\}$, $\lambda(1) = \text{focus}$). At that time, the mapping σ_{C_0} needs to map only node #1 to node #3. Below the two graphs, the row σ_{C_0} shows the mapping ($1 \rightarrow 3$ or $\sigma(1) = 3$). Suppose C_0 is extended by a **codeline** node to context C_1 . Now σ_{C_1} needs to map two nodes, but rather than starting from scratch, we assume that the previous $n - 1$ nodes have been assigned already and only the new node #2 has to be assigned. There is only one possibility in the code graph of Fig. 9, we map $2 \rightarrow 5$. In the figure σ_{C_1} corresponds to a list of length 1, with a single entry ($2 \rightarrow 5$). This entry, however, also points (vertically) to the previous assignment $1 \rightarrow 3$: Following the pointers in vertical direction reconstructs the full mapping σ_{C_1} ($1 \rightarrow 3$, $2 \rightarrow 5$). So the σ -mapping of a new context is based on an existing successor node mapping and we only supplement the assignment of the latest node to complete it.

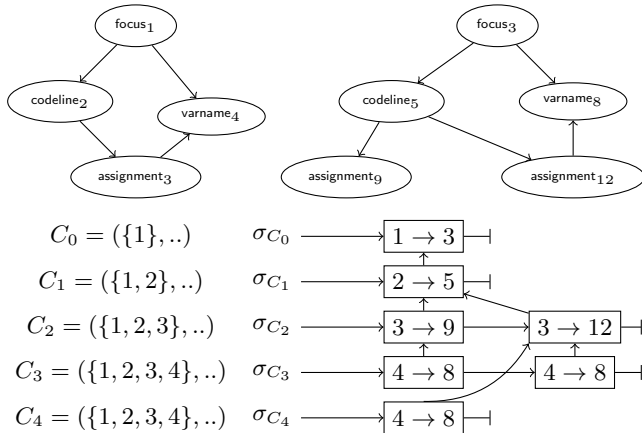


Figure 9: Efficient data structure to keep track of node assignments during the search. As a context pattern evolves (node by node) we re-use all mappings σ of successor graphs.

The main algorithm is shown in Alg. 1, which takes the set \mathcal{G} of code graphs and a start context C_0 (usually consisting of the **focus** node alone). In line 1 we initialize the search front S , which is a priority queue of limited size (we used 1000) ordered by (2). By Σ_0 we denote the set of all valid initial mappings $\Sigma_0 = \{\{\sigma_0^G\} | G \in \mathcal{G}\}$, where σ_0^G maps the unique **focus** node in the context to the unique **focus** node in the code graph G .

In line 2 we initialize a set X of explored nodes: it may happen at some point of the search that we re-consider a context graph we have visited earlier, simply because the nodes and edges of the same context graph could have been inserted in a different order. To avoid wasting time on examining graphs that were already explored, we store a hash code of all explored context graphs in X (set of explored graphs). In our implementation, we use a hash of the Weisfeiler-Lehman kernel [15] for this purpose. The remainder of Alg. 1 represents the search loop: we pick the best-so-far context graph C , expand it, check for an improvement and return the best found context C^* in the end.

Algorithm 1 ContextMining(\mathcal{G}, C_0)

```

1:  $S = \{(C_0, \Sigma_0)\}$   $\triangleright$  search front as priority queue
2:  $X = \emptyset$ ;  $(C^*, \Sigma^*) = (\perp, \perp)$   $\triangleright$  visited nodes, best context
3: while  $S \neq \emptyset$  do  $\triangleright$  queue not empty
4:    $(C, \Sigma) = \text{top}(S)$   $\triangleright$  pick best context from queue
5:    $S = \text{expand}(C, \Sigma, S, X)$   $\triangleright$  expand context
6:   if  $C^* = \perp \vee f(C) > f(C^*)$  then  $\triangleright$  improvement?
7:      $C^* = C$ ;  $\Sigma^* = \Sigma$   $\triangleright$  update best-so-far
8:   end if
9:    $X = X \cup \{h(C)\}$   $\triangleright$  save hash of explored graph
10: end while
11: return  $(C^*, \Sigma^*)$   $\triangleright$  return best context graph
```

Algorithm 2 $\text{expand}(C, \Sigma, S, X)$

```

1:  $Q = \text{explore}(C, \Sigma)$ ;  $E = \emptyset$ 
2: for  $e \in Q$  do  $\triangleright$  for all possible extensions
3:   apply extension  $e$  to context  $C$  and obtain  $C'$ 
4:   if  $h(C') \notin X \wedge \text{size}(C') < \text{limit}$  then
5:      $\Sigma' = \text{prolong-and-filter}(C', \Sigma)$ 
6:     if  $C'$  is substantial improvement of  $C$  then
7:        $C'.\text{momentum} = 2$   $\triangleright$  restore to max
8:     else
9:        $C'.\text{momentum} = C.\text{momentum} - 1$ 
10:    end if
11:    if  $f(C') > \text{worst}(S) \wedge C'.\text{momentum} > 0$  then
12:       $S = S \cup \{(C', \Sigma')\}$ 
13:    end if
14:  end if
15: end for
16: return  $S$ 
```

The node expansion is shown in Alg. 2. It calls a function $\text{explore}(\cdot)$ in line 1, which provides a priority queue (size:60) of the best possible extensions of the current context C (which includes all mentioned types of edges). This allows us to concentrate on the best extensions before we actually expand the context in the search front S . We exclude a new context if it has been explored earlier (line 4) or it becomes too large (max. 12 nodes per context). If the extended context passed all checks, we prolong or adapt the σ -mappings (line 5) to reflect newly inserted nodes. We assign a *momentum* to each context, which is initialized to a small number (here: 2) and reduced by 1 for each extension that *did not* lead to a *substantial* improvement in the object function (2). If the momentum has reached 0, the context graph will not be considered further in the search front. We use a statistical test (G-test) on the 2x2 contingency table of positive and negative cases before and after the extension to decide whether the extension was substantial.

Log mining for course recommendation in limited information scenarios

Juan Sanguino
Universidad de los Andes
Bogotá, Colombia
jc.sanguino10@uniandes.edu.co

Rubén Manrique
Universidad de los Andes
Bogotá, Colombia
rf.manrique@uniandes.edu.co

Olga Mariño
Universidad de los Andes
Bogotá, Colombia
olmarino@uniandes.edu.co

Mario Linares-Vásquez
Universidad de los Andes
Bogotá, Colombia
m.linaresv@uniandes.edu.co

Nicolas Cardozo
Universidad de los Andes
Bogotá, Colombia
n.cardozo@uniandes.edu.co

ABSTRACT

Recommender systems in educational contexts have proven effective to identify learning resources that fit the interests and needs of learners. Their usage has been of special interest in online self-learning scenarios to increase student retention and improve the learning experience. In current recommendation techniques, and in particular, in collaborative filtering recommender systems, the quality of the recommendation is largely based on the explicit or implicit information obtained about the learners. On free massive online learning platforms, however, the information available about learners may be limited and based mostly on logs from website analytics tools such as Google Analytics. In this paper, we address the challenge of recommending meaningful content with limited information from users by using rating estimation strategies from a log system. Our approach posits strategies to mine logs and generates effective ratings through the counting and temporal analysis of sessions. We evaluate different rating penalty strategies and compare the use of per-user and global metrics for rating estimation. The results show that using the average number of lessons viewed per-user is better than using global metrics with a p-value under 0.01 for 4 of our 5 hypotheses, showing statistical significance. Additionally, the results show that functions that penalize the rating to a lesser degree behave better and lead to a better recommendation.

Keywords

Recommender systems, log mining, learning trajectories

1. INTRODUCTION

The number of users enrolled in learning web platforms (*i.e.*, MOOCs) [1] has a constantly grown in the last 8 years [2]. The recent COVID-19 pandemic has reinforced this phenomenon, and web virtual learning platforms are taking on

J. Sanguino, R. Manrique, O. Mariño, M. Linares, and N. Cardozo. Log mining for course recommendation in limited information scenarios. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 430–437, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853183>

an importance not previously seen. The pandemic forced learners, course designers and instructors to migrate to virtual environments where the learning content is mainly delivered in digital formats with a limited or none face-to-face interaction. Existing work have shown the importance of simple but effective virtual learning environments, which allow a progressive adaptation according to learners needs and preferences in order to keep their motivation and engagement [3, 4].

To adapt the learning environment and give meaningful content recommendations to learners, learning platforms should be able to profile the learner in terms of a set of features such as preferences, behaviors, or learning needs. A question then arises in this context: how to obtain relevant learners' information? While more traditional approaches ask the learner directly through surveys and registration forms, modern approaches complement this explicit information with implicit data extracted from interactions with the platform (*i.e.*, logs) [5, 6]. The amount and quality of information that can be extracted depends largely on privacy regulations, the administrator guidelines, and the platform technological capabilities. Therefore it is not unusual to find open virtual learning scenarios where the information collected and available about learners is limited. This is the case for the GCFGlobal learning platform.¹

The GCFGlobal Learning Program objective is to teach different basic skills necessary for the 21st century, online, in an open modality, and without cost. According to GCFGlobal's Web page "gcfglobal.org" offers training in more than 40 topics, ranging from Microsoft Office and email usage to reading, math, and more. A GCFGlobal course contains several lessons. GCFGlobal offers more than 360 courses, counting for more than 6,400 lessons, more than 2,500 videos, and more than 50 interactive activities and games". In 2021 the number of users who used the English, Spanish and Portuguese sites and visited at least one course was around 41 million. GCFGlobal establishes as a regulatory principle of its operation that access to their learning content has to be open to anyone in the world, and the only requirement is internet access. The courses are online, self-paced and self-directed (*i.e.*, without a tutor). Registration and authenti-

¹GCFGlobal learning platform:<https://bit.ly/3tBpGa5>

cation on the platform is not mandatory and it is estimated that only 2% of users are registered, thus, there is no explicit information about learners (*i.e.*, identity data). The information that is available comes mainly from the logs via Google Analytics tool. Google Analytics is a standard and popular tool used in e-commerce that has been also positioned itself as a useful tool in learning platforms [7, 8]. Its popularity is due to its ease of implementation and its capabilities to filter and analyze large volumes of logs. Although log analysis has limitations when it comes to identifying a user, it is the most likely type of data that can be found in any existing web-based platform.

In this paper, we present an empirical study for rating estimation from Google Analytics logs. These ratings express the preferences of the learners and allow the construction of recommender systems. Our evaluation is focused on how these ratings behave in a course recommendation scenario. The contributions of our work are summarized as follows. (1) Dedicated strategies for counting lessons seen by a user, and to process the timelines from logs. (2) A strategy to generate ratings that contemplate per-user and global metrics based on lesson counting. (3) A comparison of previous strategies in a course recommendation system.

2. RELATED WORK

A recommender system in a learning platform aims to identify the most interesting learning content for learners according to their preferences [9]. The most popular recommendation techniques described in the literature are collaborative filters (CF), content-based techniques (CB), and hybrid models that combine both CF and CB [10, 11, 12, 13]. CF uses information about the rating that learners assign to courses or learning content to generate a recommendation [14, Ch.3]. CF-based systems assume that if a learner X has the same/similar rating as a learner Y on a course, X is more likely to have the same/similar rating as Y on a different course [14, Ch.3]. CB filters use a course as the basis of the recommendation, rather than a learner. That is, a CB filter uses the characteristics of the course (*i.e.*, descriptions, tags, content, topics, classifications) to make recommendations. CB filters use course characteristics to search for courses similar to those a learner has previously taken [13]. To address specific disadvantages of CF and CB models, hybrid models work on the idea of merging them to enhance the result.²

Existing work have shown the effectiveness of recommendation engines in learning platforms. For instance, Campos et al. [9] propose a CB filter that uses topic modeling for a big corpus of courses from different education platforms. Ma et al. [15] use a similar CB technique but their evaluation focuses on user experience and usability. Another common approach is to use of data mining techniques such as associate rules; for example Intayoad et al. [16] use classification algorithms and associate rules to create a system that exploits the learner’s content access history to propose recommendations.

Following the trend of e-commerce recommender systems,

²A discussion of the advantages and disadvantage of CB and CF is given by Xiao et al. [13]

there is a recent increase of hybrid models use combining both CF and CB. A representative strategy in that trend is to use a CB model to manage the problem of the cold start and CF to take advantage of the information that is produced by learners in their interaction with the platform. CB models are used to recommend items for learning content for which there is not enough information for a CF strategy. For example, a course recommendation engine can use the CB model for new students, and a CF model for existing students [13, 10]. Wang et al. [12] use a similar hybrid technique with the DCBVN framework, where the model is built with a Variational Autoencoder Network combined with a CF model to generate recommendations. DCBVN considers the course difficulty and the skills of learners to generate the recommendations.

Yin et al. [11] use three models: CF, CB (using a topic classification technique), and a transition probability model to produce a single rating for a course. The transition probability model is used to manage the relationship of prerequisites between courses. Shanshan et al. [17] use a hybrid technique to generate recommendations with a CF model and an ontology to calculate similarities. This technique uses a cascade evaluation to refine the recommendation list with a rule association algorithm.

The aforementioned models often operate in rich information scenarios. In these scenarios, demographic information, evaluation results or perception surveys are available to facilitate the construction and evaluation of the recommender system. The problem of inferring ratings is not addressed to the extent that this information is explicitly provided by the learner.

3. METHODOLOGY

In this paper, we focus on the estimation of users results and their ratings for courses via a CF recommendation technique. CF requires a rating matrix to find similar learners. However, we do not have these ratings explicitly, therefore, it is necessary to infer them from Google Analytics logs. A course developed by GCFGlobal is composed of a set of lessons. The analysis of the number of lessons accessed by the learner and the associated session time are the basis for generating the rating. In the following we present the raw data extracted from Google Analytics and its processing, as well as the rating generation strategies.

3.1 Data processing

The logs in Google Analytics are consolidated under the concept of a session. Sessions are defined as the set of events generated by user’s actions within the platform without exceeding 30 minutes of inactivity.³ Due to the way the logs layer is deployed, we have to assume that the same user accesses the platform through the same device. This statement is true in most cases and allows us to group sessions in a timeline per user. We are interested in the “PageView” event that provides the URL of the page that the user accesses. The page URLs in the learning platform are defined by a unique path for each course-lesson pair, which allows us to identify the course and lessons related to a “PageView”

³Google Analytics forum: <https://bit.ly/3IR1AAY>

event; URL paths are as “language/course/lesson/additional-params”.

We use two filters to select users and lessons from Google Analytics. The first filter removes sporadic users, keeping users with at least 30 sessions during the year (2021-01-31 to 2022-01-31). The second filter ensures that the sessions belong to at least three different courses. This filter allows us to guarantee the future construction of the dataset and ground truth to evaluate the recommendation system. We use at least two courses to build the ratings for the collaborative filter and the third one for its evaluation. It is important to mention that the temporality of the sessions is taken into account in the construction of the dataset. The first two courses in temporal order are used in the construction of the collaborative filter and the last seen was used to evaluate the recommendation. We also consider the session time factor for lesson counting as explained in the following.

After building a base of users and sessions, we proceed to perform the pre-processing steps described in Figure 1. In step 1, the base of users and sessions extracted from Google Analytics is loaded into a local database in order to be able to perform transformations on the data in an easy and efficient way. In step 2, the timestamp column is transformed and standardized in such a way that the records can be ordered from the most recent session to the most distant. As mentioned previously, this allows us to identify the chronological route that the user made through the courses and their lessons. In step 3, we identify the time spent by the user in each lesson (“lesson duration”). Step 3 is not trivial as the different sessions in which the same lesson is visited must be reconciled. An intra-lesson analysis is performed first, followed by an inter-lesson consolidation.

In the intra-session analysis, the time that a user spent in lessons of the same session is extracted. The time is calculated as the difference between the timestamp of the “PageView” event of the lesson URL and the immediately following event timestamp if it exists. In the case that the session ends with a visit to a lesson and there is no subsequent event for the calculation, we use imputation using the average lesson time over all users. For lessons with multiple visits in the same session, the times obtained were added. Table 1 shows an example of the calculation of the time spent in the lessons within the same session. Being the last event of the lesson, the time of lesson “D” is imputed using the average duration of the lesson over all users. Then an inter-lesson level analysis is carried out where the times of the lessons that were visited in different sessions are added to consolidate a total “lesson duration” per user.

Table 1: Intra session analysis example. Four “PageView” events to lessons A, B, C, and D. The time spent in lesson D is calculated as the average of all users.

Lesson	Timestamp	Time Spent (min)
A	2021-01-29 08:49:01	20
B	2021-01-29 09:09:01	18
C	2021-01-29 09:27:01	21
D	2021-01-29 09:48:01	17

In steps 4 and 5, the number of lessons that make up each course is consolidated, then, for each user-course combination, the number of lessons viewed is counted. With these results, in step 6, the percentage of lessons viewed is calculated per user-course. In step 7, the percentages obtained in step 6 are averaged to obtain a per-user metric of the percentage of lessons viewed.

In step 8, users who have seen lessons from more than 50 courses are removed. These are considered outliers because they are more than three standard deviations from the mean [18, p.19]. A total of 4 users were removed, one of them with a total of 131 courses viewed. Considering the minimum 3 courses filter explained at the beginning of this section, our dataset only contains users who have seen courses in the range (2,50).

In step 9, we add a filter to the count made in step 5 according to the user’s lesson duration. Lessons below a stipulated time will not be considered. The exclusion of lessons by duration is controlled by a parameter, and the effect of this parameter is evaluated in the experimentation. Our hypothesis is that the inclusion of this filter will allow discarding lessons that a user addressed lightly and possibly incompletely, thus improving the results of the recommendation.

Finally, in the last step (step 10), we split the data into a subset for training (*i.e.*, construction of the collaborative filter) and another for testing. The split maintains the temporal order, therefore, the test set always has courses viewed after those in the training dataset per user. We use the first 70% of the courses for building the collaborative filter and the last 30% for evaluation. In the worst case when a user has only seen three courses, two will be used for training and one for evaluation.

3.2 Rating Estimation

As mentioned above, these ratings are essential for the construction of the collaborative filter [14, Ch.2]. In this section, we explain how we estimate these ratings. Our hypothesis is that the number of lessons accessed by a user in a certain course is an expression of the level of liking of a learner for it. Being open courses where the courses and lessons are approached by the will of the learner, it seems valid to assume that the perception about a course is reflected in the number of lessons taken by the learner.

To estimate ratings we use a threshold between 0 and 5, common in recommender systems [14, Ch.2]. Courses with a percentage of lessons viewed above a threshold will obtain the maximum grade (5), and those that are below will be penalized according to a penalty function. We use two threshold metrics: (1) the average percentage of lessons viewed per course by a user (per-user metric), and (2) the overall average percentage of lessons viewed per course for all users (global metric). These two metrics are inspired by the different ways of calculating similarities [14, Ch.2], where knowing if a rating is above or below the average is considered more valuable than the rating itself. It is known that user ratings in different domains tend to be very close to the average. While a rating in the middle does not say much, a rating far from it is a clear indication of like or dislike.

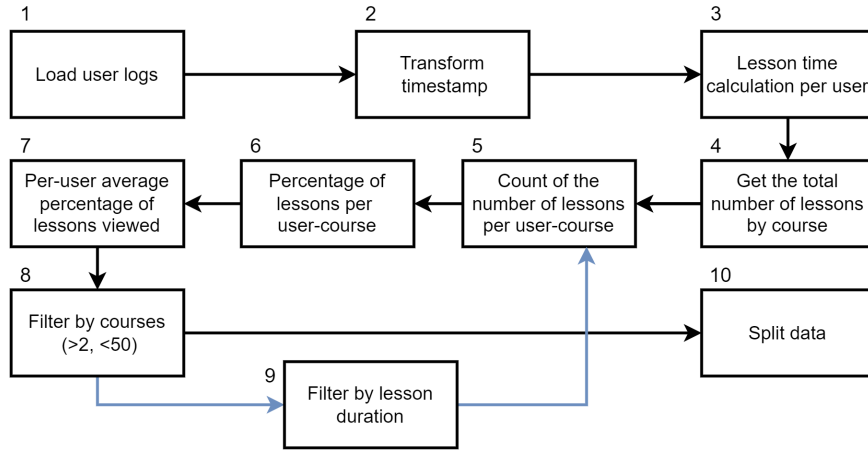


Figure 1: Data processing steps

For the rating penalty, 5 different functions will be evaluated: logarithmic, square root, quadratic, linear, and step (Equations 1 to 5). The combination of thresholds and penalty functions give us 10 different ways of calculating the ratings.

$$f_1(x) = \begin{cases} 5, & x \geq \text{threshold} \\ 5 * (\frac{x}{\text{threshold}})^2, & x < \text{threshold} \end{cases} \quad (1)$$

$$f_2(x) = \begin{cases} 5, & x \geq \text{threshold} \\ 5 * \frac{x}{\text{threshold}}, & x < \text{threshold} \end{cases} \quad (2)$$

$$f_3(x) = \begin{cases} 5, & x \geq \text{threshold} \\ 5 * \log_2((\frac{x}{\text{threshold}}) + 1), & x < \text{threshold} \end{cases} \quad (3)$$

$$f_4(x) = \begin{cases} 5, & x \geq \text{threshold} \\ 5 * \sqrt{\frac{x}{\text{threshold}}}, & x < \text{threshold} \end{cases} \quad (4)$$

$$f_5(x) = \begin{cases} 5, & x \geq \text{threshold} \\ 0, & x < \text{threshold} \end{cases} \quad (5)$$

3.3 Collaborative Filter

We use a classic collaborative filter approach since our objective is to evaluate the effect of the rating generation strategies on the recommendation as well as the temporal analysis in the filtering of the lessons.

Our collaborative filter uses the `turicreate` python library that implements an item-based CF. We use cosine similarity to identify similar courses based on ratings (Equation (6)).

$$CS(i, j) = \frac{\sum_{u \in U_{ij}} r_{ui} * r_{uj}}{\sqrt{\sum_{u \in U_i} (r_{ui})^2} * \sqrt{\sum_{u \in U_j} (r_{uj})^2}} \quad (6)$$

where U_i is the set of users who rated item i , and U_{ij} is the set of users who rated both items i and j .⁴ The library

⁴Turicreate documentation: <https://bit.ly/3y0vJsx>

has a default threshold of 0.001 where users with a lower similarity coefficient are excluded when making the rating prediction.

3.4 Evaluation setup

In step 10 of the data processing flow (Figure 1), the dataset is built using 70% of the courses for training and the remaining 30% for evaluation. Given that the average number of courses viewed by each user in our dataset is 6.64, in the average case, there are four courses for the construction of the collaborative filter and two for its evaluation.

The performance of the recommender system was evaluated by precision at k ($P@k$), a typical metric for the evaluation of Top-N recommender tasks [10, 11, 19, 20]. We found in the literature that a value of N usually used in recommender systems in learning environments is $N \in [5, 10]$ [10, 11]. A higher number of recommendations could overwhelm the learner, in particular in partially known domains of knowledge, and a lower number could include elements that are not diverse. However, the use of $k = N = 5$ for evaluation poses a challenge to the extent that in our dataset most of the users do not contain 5 relevant courses in the evaluation set. To avoid an overpenalization in the calculation of the $P@k$, we define the k per user in the range $1 \leq k \leq 5$ according to the number of courses in the test set. The results reported are the $P@K$ average over all users. A Wilcoxon rank-sum test with the Bonferroni correction was used for all our statistical significance tests ($\rho < (0.05/\#\text{hypothesis})$).

4. RESULTS

After processing the data, 56466 sessions were obtained, a total of 7071 users extracted, and 230 different courses were considered from the GCFGlobal learning platform. Table 2 presents the results obtained using the rating strategies explained in Section 3.2.

The results suggest that a custom threshold (PU: per-user) is better than a global one (GPC). However, it is not conclusive on the use of penalty functions. To identify if there are significant differences in the precision obtained by different combinations of penalty functions and threshold strategies,

Table 2: Precision results for the PU (Per-user metric) and GPC (Global metric)

Penalty Function	Threshold Strategy	Precision @ 5
F1 (Quadratic)	PU	0.3928
	GPC	0.3923
F2 (Lineal)	PU	0.3759
	GPC	0.3711
F3 (Logarithmic)	PU	0.3783
	GPC	0.3753
F4 (Square root)	PU	0.3911
	GPC	0.3824
F5 (Step)	PU	0.1677
	GPC	0.0526

we built a set composed of 100 sample. Each sample is built by selecting randomly 3000 users from the total set of users (*i.e.*, 7071). Then, we repeat the process of building and evaluating the recommendation engine. In Table 5, we present the average precision obtained per penalty function over the 100 experiments. As can be seen in the table, a similar behavior where the custom threshold (PU) and a square root function (F4) lead to a better recommendation results. Regarding the penalty functions, a very interesting behavior is evident: the lower the penalty, the better the recommendation. Figure 2 shows the penalty functions for a particular threshold from which it is possible to identify that there is a correspondence between the order in the results F4(best)-F3-F2-F1-F5(worst) and the penalty degree of the function.

We perform a statistical significance via Wilcoxon rank-sum test where the null hypothesis is that the $P@K$ obtained by the two threshold metrics have the same distribution. The alternative hypothesis is that the distribution of $P@k$ obtained by one metric is stochastically greater than the distribution of the other. We repeated this test for each penalty function, therefore a total of 5 hypotheses were evaluated. The results of the tests for each penalty function are shown in Table 3. Our first relevant finding is that the results obtained by a custom threshold (PU) are statistically significant in most cases. Only for the quadratic function, the null hypothesis was not rejected.

We repeat a similar test but for the penalty functions. Using PU as the threshold, we test if the results obtained via F4 are statistically significant in comparison with the others penalty functions. The test results are reported in Table 4 and show that results obtained via F4 are statistically significant compared to all other functions except again for the quadratic function.

Our final experimentation is related with the lesson duration (*i.e.*, Step 9 in Figure 1). We want to see the effect on the recommendation precision of adding a lesson duration filter prior to counting and calculating the percentage of lessons viewed. The filter removes lessons with a duration less than x value. We use PU as threshold metric and F4 as penalization strategy. Figure 3 shows the results obtained as

Table 3: PU (Per-user metric) vs GPC (Global metric) Wilcoxon test results.

Penalty Function	ρ -value ($\rho < (0.05/5)$)
F1 (Quadratic)	0.448
F2 (Lineal)	5.273e-32
F3 (Logarithmic)	9.649e-32
F4 (Square root)	5.750e-32
F5 (Step)	1.261e-34

Table 4: F4 vs other penalty functions via Wilcoxon test.

Penalty Functions	ρ -value ($\rho < (0.05/4)$)
F4 (Square root) vs F1 (Quadratic)	0.033
F4 (Square root) vs F2 (Lineal)	7.837e-09
F4 (Square root) vs F3 (Logarithmic)	8.400e-05
F4 (Square root) vs F5 (Step)	1.261e-34

Table 5: Average precision over 100 randomly datasets. PU: Per-user metric, GPC: Global metric

Penalty Function	Threshold Strategy	Avg. Prec. @ 5
F1 (Quadratic)	PU	0.225
	GPC	0.225
F2 (Lineal)	PU	0.239
	GPC	0.214
F3 (Logarithmic)	PU	0.241
	GPC	0.218
F4 (Square root)	PU	0.246
	GPC	0.223
F5 (Step)	PU	0.09
	GPC	0.03

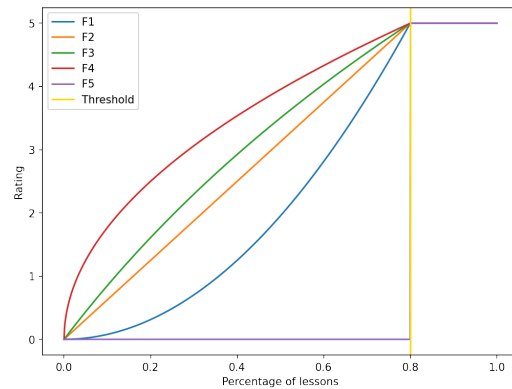


Figure 2: Behavior of penalty functions with a threshold of 0.8 (80%) for a given course. The estimated rating is 5 if the percentage of lessons accessed is greater than 80%. For lower values, the rating is given by the penalty functions F1-F5.

the filter becomes more restrictive. One of the consequences is that as the minimum lesson duration time is restricted, the

size of the dataset is reduced. This means that after applying the filter there are users with less than 3 courses and they are thus removed from the analysis. Figure 4 shows how the size of the number of users decrease with the increase of the lesson duration filter.

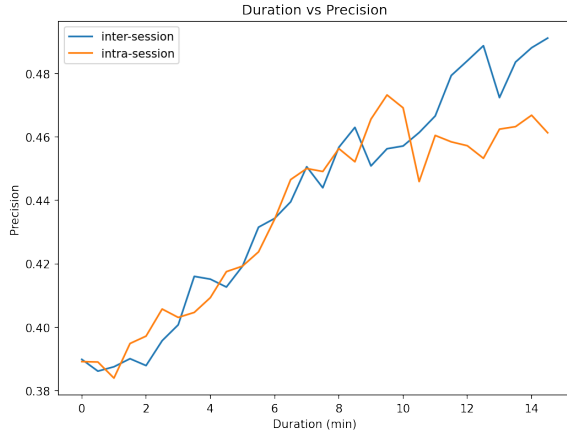


Figure 3: Lesson duration filter vs recommendation precision

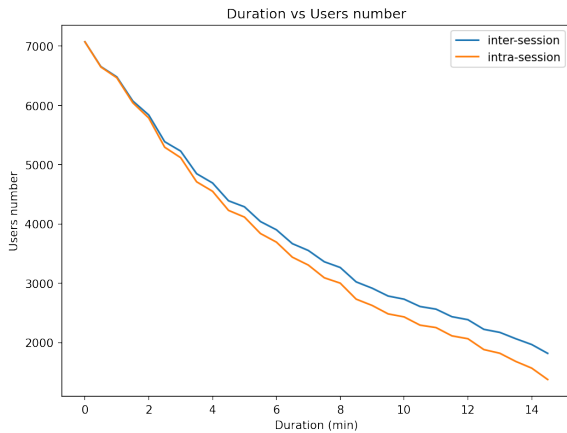


Figure 4: Lesson duration filter vs number of users in dataset

5. DISCUSSION AND FUTURE WORK

In relation to penalization functions, we observe that less penalty lead to a better recommendation. This indicates that a low number of lessons viewed per course cannot be directly related to low preference. Being open learning resources where the learning process is self-directed, there are many other variables that can affect the non-continuity in the learning process that are impossible to identify in our context (*i.e.*, learning priorities, time, health). It should also be taken into account that since it is an implicitly extracted rating, it is to be expected that it does not accurately reflect the user’s preferences. We also conclude that the F4 (square root) penalty function leads to better recommendation results, and there is statistical significance concerning F3, F2, and F5.

The experiments suggest that a per-user metric works better than a global metric. This result is not surprising to

us as using a custom metric for each user should better reflect their behavior and therefore their preferences. Despite this, keeping this metric updated is a process that can be exhausting in a productive environment, since it must be recalculated for each new interaction with a lesson. The global is expected to fluctuate less and can be recalculated every certain time interval.

Regarding the duration of the lesson, the figures show the more restrictive the filter is, the better is the recommendation precision. This behavior is independent of the way the session is processed (*i.e.*, inter vs intra). We attribute this result to the fact that the lesson duration filter reduces the data set to users for which we have (1) more information (*i.e.*, more lesson interactions), and (2) more precision in the rating calculation. As users spent more time going through the lessons, it is more likely that those lessons were of interest to them. In consequence, we can conclude that the rating estimation strategy based on course lesson completeness better reflects the preferences of learners with greater lesson duration times.

There is a wide range of future work that we want to address based on the results presented in this paper. Now that we have found an appropriate strategy to build the ratings from logs, we want to explore more advanced techniques for building the CF engine such as matrix factorization strategies and recent deep learning approaches [21, 22]. We also want to explore hybrid recommendation techniques that combine CF and CB making use, on the one hand, of textual descriptions of lessons and on the other hand, on semantic descriptions that can be enriched via knowledge graphs [23].

6. CONCLUSION

This paper addresses the problem of course recommendation under limited information scenarios. We posit a process for course rating estimation based on log information for a CF-based recommender system. Our strategy to exploit the limited information available in logs uses the combination of: (1) the percentage of the lessons accessed in a course, (2) a threshold definition, and (3) different penalty functions to estimate the learner perception about the course. Then, we evaluate the influence of the duration of a user in a lesson in the definition of the rating.

As a threshold, we found that a personalized average per-user is better than a global one. Regarding penalty functions, F4 present the best results, and in general we found that the functions that penalize to a lesser degree lead to a better recommendation. Finally, we found that “lesson duration” plays an important role to improve rating estimation. The more demanding the “lesson duration” filter better our rating estimation and the recommendation precision. However, using the lesson duration as a filter has the side effect of reducing the number of users in the dataset because a great number of lessons access records are discarded.

7. ACKNOWLEDGMENTS

The authors would like to thank GCFGlobal Learning Program for funding this project, its continuous interest in improving its learning services, and its support in consolidating the dataset used in this paper.

References

- [1] Byeongwoo Kang. How the covid-19 pandemic is reshaping the education service, 2021. URL https://doi.org/10.1007/978-981-33-4126-5_2.
- [2] Annisa R Sari, Curtis J Bonk, and Meina Zhu. Mooc instructor designs and challenges: what can be learned from existing moocs in indonesia and malaysia? *Asia Pacific Education Review*, 21:143–166, 2020. ISSN 1876-407X. doi: 10.1007/s12564-019-09618-9. URL <https://doi.org/10.1007/s12564-019-09618-9>.
- [3] Marhainis, Ismail Wan Saiful Azzam Wan, Arifin Noorfadzilah Zakaria Ibhrahim, and Jamaludin. Measuring user’s usage intentions on e-learning portal. In Janudin, Aziz Anealka, Alias Nor Aziah Luaran Johan Eddy, and Sardi, editors, *Envisioning the Future of Online Learning*, pages 347–357. Springer Singapore, 2016. ISBN 978-981-10-0954-9.
- [4] Maria José Sousa and Álvaro Rocha. Learning analytics measuring impacts on organisational performance. *Journal of Grid Computing*, 18:563–571, 2020. ISSN 1572-9184. doi: 10.1007/s10723-018-9463-1. URL <https://doi.org/10.1007/s10723-018-9463-1>.
- [5] Si Na Kew, Zaidatun Tasir, S N Kew, and Z Tasir. Learning analytics in online learning environment: A systematic review on the focuses and the types of student-related analytics data keywords learning analytics · online learning environment · student-related analytics data · intervention · educational data mining. URL <https://doi.org/10.1007/s10758-021-09541-2>.
- [6] Javaria Hassan, Jovin Leong, and Bertrand Schneider. Multimodal data collection made easy: The ez-mmla toolkit: A data collection website that provides educators and researchers with easy access to multimodal data streams. pages 579–585. Association for Computing Machinery, 2021. ISBN 9781450389358. doi: 10.1145/3448139.3448201. URL <https://doi-org.ezproxy.uniandes.edu.co:8443/10.1145/3448139.3448201>.
- [7] Daniel Brandon. Google analytics. *J. Comput. Sci. Coll.*, 34:81–82, 4 2019. ISSN 1937-4771.
- [8] Loveleen Gaur, Gurinder Singh, Jeyta, and Shubhankar Kumar. Google analytics: A tool to make websites more robust. In *Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies*. Association for Computing Machinery, 2016. ISBN 9781450339629. doi: 10.1145/2905055.2905251. URL <https://doi-org.ezproxy.uniandes.edu.co:8443/10.1145/2905055.2905251>.
- [9] Rodrigo Campos, Rodrigo Pereira dos Santos, and Jonice Oliveira. A recommendation system based on knowledge gap identification in moocs ecosystems. In *XVI Brazilian Symposium on Information Systems*. Association for Computing Machinery, 2020. ISBN 9781450388733. doi: 10.1145/3411564.3411572. URL <https://doi-org.ezproxy.uniandes.edu.co:8443/10.1145/3411564.3411572>. Tener en cuenta el esquema de evaluación, puede servir para la parte del modelo basado en contenido o en general.
- [10] Zhengjun Pan, Lianfen Zhao, Xingyu Zhong, and Zitong Xia. Application of collaborative filtering recommendation algorithm in internet online courses. In *International Conference on Big Data and Computing*, pages 142–147. Association for Computing Machinery, 2021. ISBN 9781450389808. doi: 10.1145/3469968.3469992. URL <https://doi-org.ezproxy.uniandes.edu.co:8443/10.1145/3469968.3469992>. En general presenta como generar el modelo de filtrado colaborativo mejorando su arranque en frio con un modelo basado en contenido, la entrada no es muy clara.
- [11] Shengjun Yin, Kailai Yang, and Hongzhi Wang. A mooc courses recommendation system based on learning behaviours. In *Proceedings of the ACM Turing Celebration Conference - China*, pages 133–137. ACM, 5 2020. ISBN 9781450375344. doi: 10.1145/3393527.3393550. URL <https://dl.acm.org/doi/10.1145/3393527.3393550>.
- [12] Chao Wang, Hengshu Zhu, Chen Zhu, Xi Zhang, Enhong Chen, and Hui Xiong. Personalized employee training course recommendation with career development awareness. In *Proceedings of The Web Conference*, pages 1648–1659. Association for Computing Machinery, 2020. ISBN 9781450370233. doi: 10.1145/3366423.3380236. URL <https://doi-org.ezproxy.uniandes.edu.co:8443/10.1145/3366423.3380236>.
- [13] Jun Xiao, Minjuan Wang, Bingqian Jiang, and Junli Li. A personalized recommendation system with combinatorial algorithm for online learning. *Journal of Ambient Intelligence and Humanized Computing*, 9: 667–677, 2018. ISSN 1868-5145. doi: 10.1007/s12652-017-0466-8. URL <https://doi.org/10.1007/s12652-017-0466-8>.
- [14] Charu C. Aggarwal. *Recommender Systems*. Springer International Publishing, 2016. ISBN 978-3-319-29659-3. doi: 10.1007/978-3-319-29659-3_6. URL <http://link.springer.com/10.1007/978-3-319-29659-3>.
- [15] Boxuan Ma, Min Lu, Yuta Taniguchi, and Shin’ichi Konomi. Courseq: the impact of visual and interactive course recommendation in university environments. *Research and Practice in Technology Enhanced Learning*, 16:18, 2021. ISSN 1793-7078. doi: 10.1186/s41039-021-00167-7. URL <https://doi.org/10.1186/s41039-021-00167-7>.
- [16] Wacharawan Intayoad, Till Becker, and Punnarumol Temdee. Social context-aware recommendation for personalized online learning. *Wireless Personal Communications*, 97:163–179, 2017. ISSN 1572-834X. doi: 10.1007/s11277-017-4499-2. URL <https://doi.org/10.1007/s11277-017-4499-2>.
- [17] Shang Shanshan, Gao Mingjin, and Luo Lijuan. An improved hybrid ontology-based approach for online learning resource recommendations. *Educational Technology Research and Development*, 69:2637–2661, 2021. ISSN 1556-6501. doi: 10.1007/s11423-021-10029-0. URL <https://doi.org/10.1007/s11423-021-10029-0>.
- [18] Ihab F. Ilyas and Xu Chu. *Data Cleaning*. Association for Computing Machinery, New York, NY, USA, 2019. ISBN 9781450371520.

- [19] TongXin Liao, XinHui Feng, YuanLi Sun, HongTing Wang, Cong Liao, and YuanBing Li. Online teaching platform based on big data recommendation system. pages 35–39. Association for Computing Machinery, 2020. ISBN 9781450375757. doi: 10.1145/3411681.3412951. URL <https://doi-org.ezproxy.uniandes.edu.co:8443/10.1145/3411681.3412951>.
- [20] Rubén Manrique and Olga Marino. Knowledge graph-based weighting strategies for a scholarly paper recommendation scenario. In *Workshop of Knowledge-aware and Conversational Recommender Systems RecSys*, KaRS’18, 2018. URL <https://www.w3.org/TR/sparql11-property-paths/>.
- [21] Le Wu, Xiangnan He, Xiang Wang, Kun Zhang, and Meng Wang. A survey on accuracy-oriented neural recommendation: From collaborative filtering to information-rich recommendation. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–1, 2022. doi: 10.1109/TKDE.2022.3145690.
- [22] Dheeraj Bokde, Sheetal Girase, and Debajyoti Mukhopadhyay. Matrix factorization model in collaborative filtering algorithms: A survey. *Procedia Computer Science*, 49:136–146, 2015. ISSN 1877-0509. doi: <https://doi.org/10.1016/j.procs.2015.04.237>. URL <https://www.sciencedirect.com/science/article/pii/S1877050915007462>. Proceedings of 4th International Conference on Advances in Computing, Communication and Control (ICAC3’15).
- [23] Christian Grévisse, Rubén Manrique, Olga Mariño, and Steffen Rothkugel. Knowledge graph-based teacher support for learning material authoring. In Jairo E. Serrano C. and Juan Carlos Martínez-Santos, editors, *Advances in Computing*, pages 177–191, Cham, 2018. Springer International Publishing. ISBN 978-3-319-98998-3.

Using Machine Learning Explainability Methods to Personalize Interventions for Students

Paul Hur
University of Illinois
Urbana–Champaign
khur4@illinois.edu

Suma Bhat
University of Illinois
Urbana–Champaign
spbhat2@illinois.edu

HaeJin Lee
University of Illinois
Urbana–Champaign
haejin2@illinois.edu

Nigel Bosch
University of Illinois
Urbana–Champaign
pnb@illinois.edu

ABSTRACT

Machine learning is a powerful method for predicting the outcomes of interactions with educational software, such as the grade a student is likely to receive. However, a predicted outcome alone provides little insight regarding how a student’s experience should be personalized based on that outcome. In this paper, we explore a generalizable approach for resolving this issue by personalizing learning using explanations of predictions generated via machine learning explainability methods. We tested the approach in a self-guided, self-paced online learning system for college-level introductory statistics topics that provided personalized interventions for encouraging self-regulated learning behaviors. The system used explanations generated by SHAP (SHapley Additive exPlanations) to recommend specific actions for students to take based on features that most negatively influenced predicted learning outcomes; an “expert system” comparison condition provided recommendations based on predefined rules. A randomized controlled trial of 73 participants (37 expert-system condition, 36 explanation condition) revealed similar learning and topic-choosing behavior between conditions, suggesting that XAI-informed interventions facilitated student statistics learning to a similar degree as expert-system interventions.

Keywords

Machine Learning Explainability, Online Learning, Self-regulated Learning, Educational Interventions

1. INTRODUCTION

Personalization promotes learning by providing meaningful, timely, and relevant support that is tailored and paced to an individual’s needs and preferences [4, 32]. Thus, many intelligent tutoring systems (ITSs) have integrated person-

alization aspects that can automatically suggest which materials to study [9], reorient attentional states [15], and construct personalized feedback [20]. Such interventions are often driven by predictions using past learners’ data to build machine learning models, whose underlying mechanisms can be difficult to interpret. Yet, understanding the reasons behind a prediction is essential for educational software that needs to respond not only to *what* is likely (i.e., predicted) to happen, but also *why* it is likely.

Explainable artificial intelligence (XAI) methods [8, 17] have been developed to circumvent the opaque nature of complex machine learning models, which may thus enable a new generation of educational software with increased user trust and perceived usefulness [12, 13]. In this paper, we create personalized interventions driven by explanations, rather than by predictions, for the purpose of adapting students’ behaviors in a computer-based learning environment. We focus on encouraging self-regulated learning (SRL) behaviors in particular [33]. SRL is especially important in online and computer-based learning contexts, where teachers are often less available (versus classroom learning contexts) to guide the learning process. However, many students need assistance with these SRL decisions [37, 45, 36], and thus stand to benefit from computer-based learning environments that fill the gaps in SRL skills by suggesting appropriate activities to students.

We present work from a randomized controlled trial for which we developed an online, computer-based education platform for college-level introductory statistics topics. We explored how machine learning model predictions, coupled with explanations, can personalize interventions to support SRL reviewing behaviors. Our study provides a rigorous comparison of an XAI-driven intervention against an active expert-system intervention consisting of predefined rules based on the amount of time spent studying each topic and the expected order of topics in the curriculum.

The XAI-driven interface adaptations in this work raise several research questions (RQs) related to the effect interventions have on learning and the effects adaptations have on behaviors. These RQs have implications for computer-based education (and for broader understanding of how simple

P. Hur, H. Lee, S. Bhat, and N. Bosch. Using machine learning explainability methods to personalize interventions for students. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 438–445, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853181>

XAI-driven interventions affect student behaviors).

RQ1: What are the effects on learning and self-regulated learning behaviors when students receive XAI-informed interventions vs. expert-system interventions?

Hypothesis: We expected that the participants in the XAI-informed intervention group will learn more than those receiving expert-system interventions due to XAI-informed intervention group studying topics directly related to improving their predicted learning outcome. Furthermore, we expected that XAI-informed interventions would lead to more frequent reviewing SRL behaviors. Although both the XAI-informed and expert-system condition interventions in our study were created with the goal of supporting SRL reviewing behaviors (re-taking quizzes and re-reading texts), we expected that XAI-informed interventions would better highlight topic areas that needed the most studying based on learning outcome predictions.

RQ2: Do XAI-informed interventions lead to different topic choosing behaviors compared to the expert-system intervention?

Hypothesis: XAI-informed interventions may impart topic choosing strategies based on predicted knowledge gaps, which we expected would lead to lower proportions of students following the default intended order of topics. Furthermore, we expected that students would follow interventions in the expert-system condition more frequently because these interventions often recommended a top-to-bottom reading order of topics that may align with students' natural inclinations.

2. RELATED WORK

Here, we highlight work on self-regulated learning in online and computer-based education environments and interface adaptations informed by XAI.

2.1 Supporting Self-Regulated Learning

Self-regulated learning (SRL) refers to the metacognitive, motivational, and emotional processes behind acquiring information or skills [48, 33]. SRL has been identified as an important skill for succeeding in postsecondary education [27, 31]. Developing SRL skills is difficult, however; students struggle to differentiate effectiveness between learning strategies [50], and may not be aware of how to develop SRL skills [5]. There are three general groups of strategies identified in major SRL models: preparation, performance, and regulation [33, 10, 38, 44, 50, 38, 44] (though there is some variation, including SRL strategies that occur after/between learning sessions [49]). It is regulation behaviors (e.g., re-visiting materials or re-taking quizzes to prepare for a final test) our study interventions target, since reviewing behaviors can be supported by recommending review of specific learning material during test preparation.

Within the past two decades, a substantial amount of work has been carried out encouraging SRL in online learning environments, such as MOOCs [25, 23], where SRL skills may be especially important since learners are required to learn autonomously [2, 42]. Researchers have developed computer-based education environments to support SRL skills,

such as MetaTutor [3], Betty's Brain [24], and Cognitive Tutor [40], which aid SRL skills via adaptive pedagogical agents or by automatically personalizing the presentation of information. These systems, along with other research in online contexts [14, 34], demonstrate the feasibility of utilizing data recorded in log files to examine SRL behaviors through modeling SRL behaviors and predicting student outcomes.

2.2 Adaptations using XAI

In this paper, we focus on a particular XAI method called SHAP (SHapley Additive exPlanations) [26]. SHAP is well-suited to driving interface adaptations because it provides, for every prediction, an indication of how much each feature (i.e., predictor variable) influenced the decision made by a machine learning model. SHAP values capture directionality (e.g., the value of feature X_1 for this prediction contributed positively vs. negatively to the prediction) as well as magnitude, via a game-theoretic approach [22]. Hence, an interface can adapt to the needs of users based on feature values and the effects those values have on predictions (i.e., the SHAP values), provided that the features themselves are interpretable [6].

Within XAI research, there has been less focus on XAI systems that leverage machine learning explainability for adaptation for education purposes. Conati et al. used XAI for integrating explanation functionality for adaptive hints in an Adaptive CSP (ACSP) [12], and found that explanations increase students' trust, perceived usefulness, and intention to use the hints again. In another study by Mu [30], researchers used XAI to develop suitable interventions for wheel-spinning students with simulated data and hypothetical interventions predicted for a previous study [30]. The work in our paper significantly extends this previous work [30] by examining one possible application of XAI-driven interventions (i.e., supporting SRL behaviors) via a randomized controlled trial. We also explore how XAI approaches such as SHAP can help education researchers discover sensible interventions for any learning behavior (e.g., suggest different things to different students in a plausible way)—in our case, SRL reviewing behaviors.

3. METHODS

Next, we discuss our online learning system and SRL interventions, the machine learning model for predicting student learning outcome and interpretation via SHAP (SHapley Additive exPlanations) values, and the experiment setup.

3.1 Self-guided Online Learning System

We developed a self-guided, self-paced online learning system which displays both learning content and interventions as students navigated through the interface, agnostic of content type (images, text, videos, etc.). The system also collected logs including some general interaction behaviors such as web page visits, time spent on each page, and more specific study-related data such as automatically assessed quiz scores, pretest scores, and posttest scores.

We focused on introductory statistics because it is an important yet difficult-to-learn subject for many college degrees [46, 39, 41]. We developed a small curriculum of 12 introductory statistics topics in consultation with university

statistics instructors and educational websites (e.g., course pages). Each topic consisted of a reading (text tutorials and accompanying figures) and a corresponding 3-question, multiple choice mini quiz. These materials could be accessed from the main interface of the topics menu page (Fig. 2). The curriculum also included two variations of a 12-question multiple choice question test (a pretest and a posttest), with each question of the tests corresponding directly to one of the 12 topics. Both tests asked about the same core concepts and differed only with slight variations in questions, such as the specific values used. We designed the final curriculum to take a total of 90 minutes to complete, including the pretest, 12 readings, 12 quizzes, and the posttest.

3.2 Expert-system Intervention

We designed an expert-system version of the self-regulated learning intervention for the system (Fig. 1, top image) with a simple yet precise message of a topic suggestion based on reading time. However, we decided that the expert-system intervention should first suggest an unseen topic over topics with little study time, since learning outcomes improve when students at least touch on all material in time-limited scenarios [28]. We anticipated that the statistics topics on the system required careful reading in order to fully learn and perform well, as the study’s statistics topics have been cited to be prone to misconception [43], and difficult to teach [11]. Thus, we expected that time spent on readings was closely connected to posttest scores. If a student knew that they had spent a lower amount of time reading one of the topics relative to others, they may self-reflect and be more likely to prioritize reviewing that topic over others they have already studied more thoroughly.

We implemented the expert-system intervention in the on-line learning system by displaying the intervention message when the student reached the 30 minute mark in the self-guided study session (Fig. 2), then again at 40 minutes, and finally, at 50 minutes. We chose these time points to provide sufficient data collection before the first intervention to enable an accurate prediction of student outcome, and repeated the intervention at 10-minute intervals to give the students additional suggestions. At the 60 minute mark, students were automatically taken to the posttest. Since the study session was self-guided, it was left up to the discretion of the student to read, review, or skip topics, and spend as little or as much time—up to 60 minutes—as they wished to complete the study session.

3.3 Piloting and Training Data Collection

Please study this topic next (may be helpful to write down the topic name): Summarizing Qualitative Data

This recommendation is based on your reading time so far. Focusing on this topic may help you learn a topic you have not studied as much.

Please study this topic next (may be helpful to write down the topic name): Probability Introduction

This recommendation is based on a prediction of which topic is most likely to help you on the test at the end. Focusing on this topic may help you learn a topic you have not yet mastered as well as the others.

Figure 1: Examples of the expert-system (top) vs. XAI-informed (bottom) intervention messages.

TOPICS MENU

► Why Learn Statistics? (Click to show/hide)

Descriptive Statistics

You may choose sections from any chapter in any order. You will find instructional texts (in green) as well as questions (in blue) associated with each reading below.

- Read Types of Data - Text
- Answer Questions Types of Data - Questions
- Read Summarizing Qualitative Data - Text
- Answer Questions Summarizing Qualitative Data - Questions
- Read Summarizing Quantitative Data - Text
- Answer questions Summarizing Quantitative Data - Questions
- Read Measures of Variability - Text
- Answer questions Measures of Variability - Questions

Figure 2: A portion of the topics menu from a self-paced learning session.

We recruited student participants via student mailing lists and digital bulletin boards, seeking students with minimal college statistics experience (0 or 1 college-level statistics courses) in order to avoid ceiling effects from participants with extensive preexisting knowledge of the material. The study session was fully online. The study included a demographics survey, a pretest, a self-guided learning session (12 readings and 12 quizzes), and a final posttest. Participants were compensated \$15 USD. Based on participant feedback from semi-structured interviews (compensated an additional \$5), we made various minor changes, such as clarifying the topics menu page instructions, adjusting names to reduce cultural specificity, noting topics from the topics menu were related to the pretest and posttest questions, and including a proceed to posttest confirmation page.

After making the final changes to our system, we recruited a total of 58 participants for the first round of data collection. The goal of the first round of data collection was to collect training data for the machine learning model to predict the posttest score, which is described in the next section.

3.4 XAI-informed Intervention

Table 1: Example subset of SHAP values from a posttest score prediction for one student, indicating that the student’s current time spent on topic 8 (i.e., 0 seconds) has the most negative impact on their predicted posttest outcome.

Feature name	Feature value	SHAP value
Pretest score	50%	2.454
Quiz 1 score	67%	0.206
Quiz 2 score	100%	0.221
...
Topic 6 reading time	658 seconds	3.734
Topic 7 reading time	0 seconds	-3.860
Topic 8 reading time	0 seconds	-6.187

With the training data collected with 58 participants, we trained a random forest regressor using pretest score, quiz

score (12 topics), and reading time (12 topics) features to predict posttest score. We trained the final model on all data with 100 trees and a maximum tree depth of 4 (the only hyperparameter tuned). We used the tree explainer (`shap.TreeExplainer`) in the Python `shap` library [26] to interpret model behavior of the posttest score predictions in terms of SHAP values. The feature with the most negative SHAP value represented the feature which contributed most to lowering a student’s posttest score.

For the XAI-informed intervention message (Figure 1, right image) used for the XAI-informed condition, the topic (quiz or reading) with the most negative SHAP value was selected to be recommended to the student. The intervention text also communicated to the student that the recommendation was based on a system prediction of the student’s posttest score to help the student better understand the reason for suggesting the particular topic.

We examined what happens when a student follows the XAI-informed intervention recommendation of the feature with the most negative SHAP value. Table 1 shows an excerpt of results from a SHAP analysis of a student during a study session. At the time of this particular prediction, *Topic 8 reading time* had the most negative SHAP value with -6.187—substantially lower than the next feature, *Topic 7 reading time* at -3.860. This indicates that, based on the model’s posttest prediction, topic 8 reading time is negatively impacting the predicted posttest score of this student by 6.187 points out of 100 possible points on the posttest, and should be recommended to the student to study.

Figure 3 shows how the SHAP value of one example feature, *Topic 6 reading time*, changed with each additional 20 seconds of studying time for one student. The SHAP value trended in the positive direction as reading time increased until a point between around 140 seconds of total reading time when the SHAP value plateaued at ≈ 4 . The model appears to have learned that short studying times do not yield learning, and that very long studying times (relative to the brief topics used in this experiment) do not help past a certain point, thus yielding an approximately sigmoid-shaped curve. In this example, what was previously negatively influencing the predicted posttest score is now predicted to contribute around +4 points toward the final posttest score. Using the intuition from this example, our XAI-informed intervention recommends a topic to review to the student such that each student receives a personalized recommendation of the most helpful topic for improving their posttest score.

3.5 Expert-system vs. XAI-informed Intervention Experiment

We carried out a randomized controlled trial to compare the expert-system intervention and the XAI-informed intervention. We recruited 73 participants with minimal college statistics experience (0 or 1 college-level statistics courses) via campus mailing lists and—with the aid of university research support—targeted emails to undergraduate students with no statistics course on their academic course record. We also recruited students from research subject pools and introductory psychology undergraduate courses.

We randomly assigned students to conditions, with 37 stu-

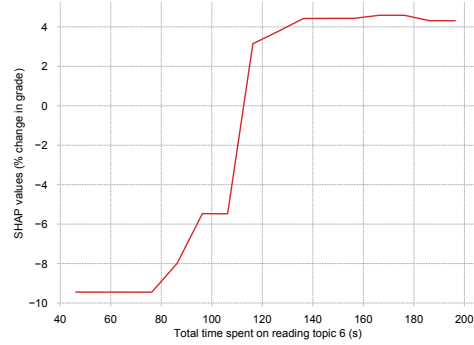


Figure 3: Changes in student’s total time spent on reading topic 6 and resulting SHAP values (in % change in grade)

dents and 36 students assigned to the expert-system and XAI-informed conditions, respectively. The study session structure was identical to the training data study sessions: a video-call meeting followed by the student independently working through the study consisting of a demographics survey, a pretest, a self-guided learning session (12 readings and 12 quizzes), and a final posttest. Students were told that the estimated total completion time was around 90 minutes, and were compensated \$15 USD. However, for the experiment, in the case that a participant wanted to skip out of the self-guided learning session early and proceed to the posttest, they would receive an intervention on the confirmation page and be offered the opportunity to return to the learning session. We included this final intervention to ensure that every student saw at least one intervention message regardless of whether they reached the 30 minute mark.

4. RESULTS

In this section, we report participant demographics, various learning outcome comparisons, and finer-grained analyses of participants’ topic ordering learning behaviors.

4.1 Demographics Information

Among the 73 participants from the randomized control trial, 73% identified as female, 26% as male, and 1% as non-binary. Students had a mean age of 19.58 ($SD = 1.71$) years old, with a minimum age of 18 and a maximum of 27. Over 35 college majors were represented by our participant population. Finally, 55% identified as White, 27% Asian, 12% Hispanic or Latina/o, 3% Black/African American, and 1% Native American.

4.2 Learning Behaviors and Outcomes

Table 2 summarizes the differences in pretest and posttest scores between the expert-system and XAI-informed groups. The mean improvement from pretest to posttest score was 18.03 (out of 100) for the expert-system condition, and 10.88 for the XAI-informed conditions, suggesting—contrary to RQ1 expectations—that students in the expert-system condition may have learned more. However, the difference in improvement between the two conditions was not significant, $t(71) = 1.924$, $p = .058$. We also calculated the Bayes factor (BF) using JASP [19]; BF represents how

likely the null or alternate hypothesis model is through a Bayesian approach [18]. Established guidelines [18] suggest that $BF = 1-3$ provides anecdotal evidence and $BF = 3-10$ provides substantial evidence. Through this metric, there is anecdotal evidence that the expert-system intervention group had greater grade improvement, $BF = 2.24$. Table 2 also shows that both groups' mean scores were not likely to have been influenced by ceiling effects from prior statistics knowledge; furthermore, no student in either condition achieved a perfect score (100) on the pretest.

Table 2: Comparison of pretest and posttest scores between expert-system and XAI-informed conditions

Group	Count	Mean score	Std. dev.
Expert-system (pretest)	37	47.1%	16.10
XAI-informed (pretest)	36	50.2%	19.87
Expert-system (posttest)	37	65.1%	16.30
XAI-informed (posttest)	36	61.1%	20.70

4.3 Model Evaluation and XAI-informed Predictions

We performed 5-fold cross-validation with the model training data to estimate model accuracy and obtained a mean R^2 value of .262 ($SD = .067$), a mean RMSE of 15.17 (on a 0-100 posttest grade scale, $SD = 2.20$), and a mean Pearson's r of .576. Mean R^2 value was somewhat variable across cross-validation folds; however, the trained model worked relatively well overall when considering the small size of our training data.

We analyzed the predictions made by our model and the activity logs of the XAI-informed intervention group participants. We found that the top three most often recommended topics were *Probability Introduction*, *Introduction to Regression*, and *Calculating Probability*. These represent the most frequently recommended topics that had the most negative SHAP values at the time of displaying the intervention, across the 30 minute, 40 minute, 50 minute mark, and on the posttest confirmation page. Two of the three most-recommended topics were related to probability. Probability is widely recognized as a difficult topic to learn for students due to misconceptions about the subject [7, 16, 21, 43], and our findings here support this assertion.

4.4 Self-Regulated Learning Behaviors

In order to evaluate the effects of the interventions on self-regulated learning behaviors, we defined two metrics which are shown in Table 3: attempts at quizzes already taken and rereading texts that had already been read. The differences in the number of quiz retakes were not significant, $t(71) = -1.618$, $p = .110$, but there was anecdotal evidence that the XAI-intervention group did more quiz retakes, $BF = 1.354$. Similarly, the number of text reviews was not significantly different, $t(71) = -1.186$, $p = .240$, but there was anecdotal evidence of the XAI-intervention group having higher number of text reviews, $BF = 2.263$. The mean number of interventions seen prior to the posttest was 3.16 for the XAI-informed intervention group, and 2.75 for the

expert-system intervention group. However, there were several participants who only saw a single intervention: 10 from the XAI-informed intervention group and 3 from the expert-system intervention group.

The results in Table 3 suggest minimal reviewing behaviors in both conditions, though that may be expected given that learning session included enough content that students could spend most or all of their time on new topics.

Table 3: Comparison of metrics for SRL reviewing behaviors.

Group	Quiz retakes	Texts reread
Expert-system	1.054	11.514
XAI	1.583	15.444

4.5 Learning Order Analysis

We carried out an analysis to examine orders in which students in each condition studied the 12 topics. We analyzed the degree to which students deviated from the baseline learning topic order by calculating the proportion of topic component (reading or quiz) selection actions that did not follow the direction of the default learning order presented on the topics menu page (Fig. 2). For example, if the student studied the first three topics in order and then studied the sixth topic, the learning order deviation value for readings would be .333. These learning order deviation values were calculated for the learning periods before and after students saw the first intervention. This analysis was done for participants who studied any amount of material after the first intervention $n(\text{XAI}) = 26$, $n(\text{expert-system}) = 34$.

The results in Table 4 show that students in both conditions deviated from the typical top-to-bottom topic significantly more frequently after intervention: XAI-informed, $t(25) = 4.262$, $p < .001$; expert-system, $t(33) = 3.240$, $p = .003$. These differences before and after the intervention were expected, per RQ1, especially for the XAI-informed intervention condition since it is more likely to recommend topics out of order according to students' individual needs.

Table 4: Proportion of actions in which students deviated from a typical top-to-bottom topic order during their selections of what topic to pursue next.

	Topic order deviation	
	Expert-system	XAI
Before 1st intervention	.552	.539
After 1st intervention	.709	.727
Difference before/after	.157	.188

5. DISCUSSION

Here, we discuss the main findings and implications and also discuss generalization of our approach, limitations of our study, and possible future work.

5.1 Learning and SRL Behaviors (RQ1)

We hypothesized that the students in the XAI-informed condition would have greater learning gains when compared

to those receiving the expert-system intervention because XAI interventions would give suggestions to review the most critical topics for improving posttest score rather than the expert-condition suggestions based on time. However, the findings did not support our hypothesis. Learning gain was not significantly different between the two conditions. The overall average pretest score being 48.6% and average posttest score 63.1%, and thus students may have benefited from studying almost *any* topic. In such cases, an intervention to encourage specific SRL behaviors would not be needed until the student has spent much longer studying.

Additionally, our machine learning model was trained from a relatively small amount of training data of 58 students, which may have contributed noise to the predictions, and consequently, less effective XAI-informed interventions which recommended unhelpful topics for improving the posttest score. However, both the expert-system and XAI-informed groups had significant, notable improvements in pretest to posttest scores, showing that our curriculum (and perhaps both interventions) was effective for teaching the statistics topics.

We hypothesized that the XAI-informed interventions would have lead to more frequent SRL reviewing behaviors due to bringing to light more directed learning strategies motivated by improving one's posttest score, and therefore, have lead to more regular and frequent self-reflection to identify apparent gaps in learning. The findings were inconclusive for answering our hypothesis since there were few instances of SRL reviewing behaviors (Table 3). While there were no statistical differences between the mean quiz and text reviewing behaviors, Bayes factor values show that there was anecdotal evidence of the XAI-informed group having both slightly higher mean rates of retaking quizzes and rereading texts. This suggests that there may indeed be intervention effects that could emerge more clearly in large datasets and longer learning sessions.

5.2 Topic Choosing Behaviors (RQ2)

We expected that participants receiving the expert-system interventions would have been more receptive to following the interventions' suggestions compared to those receiving the XAI-informed interventions. The expert-system interventions recommended topics based on the lowest reading time spent, or to suggest the next unstudied topic in the expected order. This would have aligned with some students' natural inclinations of studying through the topics in the default order as presented on the topics menu page (Figure 2). Furthermore, we expected the XAI-informed intervention group to strategize their topic choosing behavior without the influence of a necessarily top-to-bottom order suggestion, and choose more autonomously, based on strategies of identifying gaps in knowledge (discussed in section 5.1).

The results show that there were increased effects on topic choosing deviation behaviors in the two conditions. XAI-informed and expert-system conditions both had increased proportions of topic selection behaviors deviating from the top-to-bottom order after the first intervention. The results suggest the possibility that both interventions facilitated self-directed learning behaviors of students to make topic choosing behaviors most beneficial for their learning.

When understanding the results of RQ1 on learning performance and topic suggestions together, the findings from our paper may suggest that the XAI-informed interventions facilitated students' statistics learning to a similar degree to a method which prioritized unseen topics over mastery of a smaller amount of material. Additionally, average pretest scores were quite low for both conditions (Table 2), which likely made any amount of scaffolding helpful.

5.3 Generalization to Other Domains

In our study, we used the possible reasons from quizzes and readings for the posttest score predictions to personalize interventions to scaffold SRL behaviors through encouraging the review of specific topics. However, the same XAI approach could be applied to cases where the goal is to help a student improve almost any predicted outcome. For example, one could predict student dropout from online learning based on relevant student factors (constructs related to study skills, learning material interaction patterns, stress, motivation, etc.) [35], combined with complex or uninterpretable factors, and implement XAI-informed alerts via interfaces or targeted emails for the student throughout the course period to take actions reducing the likelihood of dropout based on the most impactful of the interpretable factors. Other applications could follow a similar approach, such as helping students resolve confusion during attempting algebra assignments [1], preventing learner disengagement from reading texts [29], or improving student performance in interactive online question pools [47].

5.4 Future Work and Conclusion

Future work could expand on this research by increasing the number of participants in both the training data and experiment as we were limited by our relatively small pilot and experiment sample sizes. It may also be possible to reduce idling behavior instances by using pre-screening surveys to gauge their motivation for learning the material, or better controlling the experiment through an in-person lab setting where participants may be observed. It would also be informative to explore our approach to other types of study material, such as more advanced statistics topics or mathematics, and explore ways to support other SRL behaviors such as planning or goal-setting.

In this study, we leveraged machine learning to predict future student outcomes, explained the predictions via an XAI method, and implemented personalized system interventions. Specifically, we explored supporting SRL behaviors in an online learning environment for learning college-level introductory statistics topics through personalized interventions. Despite limited differences in learning gain, SRL reviewing behaviors, and topic choosing behaviors, our findings suggest that XAI-informed interventions facilitate learning to a similar benefit as expert-system interventions. We expect that the approach examined in this experiment could be generalized across other applications, and could serve as one reference for designing system implementation informed by XAI methods.

6. ACKNOWLEDGEMENTS

This research was supported by a grant from the Technology Innovation in Educational Research and Design (TIER-ED) initiative at the University of Illinois Urbana-Champaign.

7. REFERENCES

- [1] S. M. R. Abidi, M. Hussain, Y. Xu, and W. Zhang. Prediction of confusion attempting algebra homework in an intelligent tutoring system through machine learning techniques for educational sustainable development. *Sustainability*, 11(1):105, 2019.
- [2] M. E. Alonso-Mencía, C. Alario-Hoyos, J. Maldonado-Mahauad, I. Estévez-Ayres, M. Pérez-Sanagustín, and C. Delgado Kloos. Self-regulated learning in moocs: lessons learned from a literature review. *Educational Review*, 72(3):319–345, 2020.
- [3] R. Azevedo, A. Johnson, A. Chauncey, and C. Burkett. Self-regulated learning with metatutor: Advancing the science of learning with metacognitive tools. In I. M. Khine, Myint Sweand Saleh, editor, *New science of learning*, pages 225–247. Springer, New York, NY, 2010.
- [4] M. L. Bernacki, M. J. Greene, and N. G. Lobczowski. A systematic review of research on personalized learning: Personalized by whom, to what, how, and for what purpose (s)? *Educational Psychology Review*, 33(4):1675–1715, 2021.
- [5] R. A. Bjork, J. Dunlosky, and N. Kornell. Self-regulated learning: Beliefs, techniques, and illusions. *Annual review of psychology*, 64:417–444, 2013.
- [6] N. Bosch. AutoML feature engineering for student modeling yields high accuracy, but limited interpretability. *Journal of Educational Data Mining*, 13(2):55–79, 2021.
- [7] K. Carolyn and S. Kirk. A new approach to learning probability in the first statistics course. *Journal of Statistics Education*, 9(3), 2001.
- [8] D. V. Carvalho, E. M. Pereira, and J. S. Cardoso. Machine learning interpretability: A survey on methods and metrics. *Electronics*, 8(8):832, 2019.
- [9] N. Chakraborty, S. Roy, W. L. Leite, M. K. S. Faradonbeh, and G. Michailidis. The effects of a personalized recommendation system on students’ high-stakes achievement scores: A field experiment. *International Educational Data Mining Society*, 2021.
- [10] A. Cicchinelli, E. Veas, A. Pardo, V. Pammer-Schindler, A. Fessler, C. Barreiros, and S. Lindstädt. Finding traces of self-regulated learning in activity streams. In *Proceedings of the 8th international conference on learning analytics and knowledge*, pages 191–200, 2018.
- [11] G. W. Cobb and D. S. Moore. Mathematics, statistics, and teaching. *The American mathematical monthly*, 104(9):801–823, 1997.
- [12] C. Conati, O. Barral, V. Putnam, and L. Rieger. Toward personalized xai: A case study in intelligent tutoring systems. *Artificial Intelligence*, 298:103503, 2021.
- [13] C. Conati, K. Porayska-Pomsta, and M. Mavrikis. Ai in education needs interpretable machine learning: Lessons from open learner modelling. *arXiv preprint arXiv:1807.00154*, 2018.
- [14] S. Crossley, M. Dascalu, D. S. McNamara, R. Baker, and S. Trausan-Matu. Predicting success in massive open online courses (moocs) using cohesion network analysis. Philadelphia, PA: International Society of the Learning Sciences., 2017.
- [15] S. D’Mello, A. Olney, C. Williams, and P. Hays. Gaze tutor: A gaze-reactive intelligent tutoring system. *International Journal of human-computer studies*, 70(5):377–398, 2012.
- [16] J. Garfield and A. Ahlgren. Difficulties in learning basic concepts in probability and statistics: Implications for research. *Journal for research in Mathematics Education*, 19(1):44–63, 1988.
- [17] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi. A survey of methods for explaining black box models. *ACM computing surveys (CSUR)*, 51(5):1–42, 2018.
- [18] A. F. Jarosz and J. Wiley. What are the odds? a practical guide to computing and reporting bayes factors. *The Journal of Problem Solving*, 7(1):2, 2014.
- [19] JASP Team. JASP (Version 0.16.1)[Computer software], 2022.
- [20] E. Kochmar, D. D. Vu, R. Belfer, V. Gupta, I. V. Serban, and J. Pineau. Automated personalized feedback improves learning gains in an intelligent tutoring system. In *International Conference on Artificial Intelligence in Education*, pages 140–146. Springer, 2020.
- [21] C. Konold. Issues in assessing conceptual understanding in probability and statistics. *Journal of statistics education*, 3(1), 1995.
- [22] I. E. Kumar, S. Venkatasubramanian, C. Scheidegger, and S. Friedler. Problems with Shapley-value-based explanations as feature importance measures. In *Proceedings of the 37th International Conference on Machine Learning*, pages 5491–5500. PMLR, Nov. 2020.
- [23] D. Lee, S. L. Watson, and W. R. Watson. Systematic literature review on self-regulated learning in massive open online courses. *Australasian Journal of Educational Technology*, 35(1), 2019.
- [24] K. Leelawong and G. Biswas. Designing learning by teaching agents: The betty’s brain system. *International Journal of Artificial Intelligence in Education*, 18(3):181–208, 2008.
- [25] A. Littlejohn, N. Hood, C. Milligan, and P. Mustain. Learning in moocs: Motivations and self-regulated learning in moocs. *The internet and higher education*, 29:40–48, 2016.
- [26] S. M. Lundberg, G. Erion, H. Chen, A. DeGrave, J. M. Prutkin, B. Nair, R. Katz, J. Himmelfarb, N. Bansal, and S.-I. Lee. From local explanations to global understanding with explainable ai for trees. *Nature machine intelligence*, 2(1):56–67, 2020.
- [27] C. Mega, L. Ronconi, and R. De Beni. What makes a good student? how emotions, self-regulated learning, and motivation contribute to academic achievement. *Journal of educational psychology*, 106(1):121, 2014.
- [28] P. Michlík and M. Bieliková. Exercises recommending for limited time learning. *Procedia Computer Science*, 1(2):2821–2828, 2010.
- [29] C. Mills, N. Bosch, A. Graesser, and S. D’Mello. To quit or not to quit: predicting future behavioral disengagement from reading patterns. In *International Conference on Intelligent Tutoring Systems*, pages

- 19–28. Springer, 2014.
- [30] T. Mu, A. Jetten, and E. Brunskill. Towards suggesting actionable interventions for wheel-spinning students. In *Proceedings of the 13th International Conference on Educational Data Mining (EDM 2020)*, pages 183–193. International Educational Data Mining Society, July 2020.
- [31] W. C. Naumann, D. Bandalos, and T. B. Gutkin. Identifying variables that predict college success for first-generation college students. *Journal of College Admission*, (181):4, 2003.
- [32] U. S. D. of Education. Transforming american education: Learning powered by technology. 2010.
- [33] E. Panadero. A review of self-regulated learning: Six models and four directions for research. *Frontiers in psychology*, 8:422, 2017.
- [34] A. Pardo, F. Han, and R. A. Ellis. Combining university student self-regulated learning indicators and engagement with online learning events to predict academic performance. *IEEE Transactions on Learning Technologies*, 10(1):82–92, 2016.
- [35] J.-H. Park. Factors related to learner dropout in online learning. *Online Submission*, 2007.
- [36] M. Pedrotti and N. Nistor. How students fail to self-regulate their online learning experience. In *European Conference on Technology Enhanced Learning*, pages 377–385. Springer, 2019.
- [37] N. E. Perry, L. Hutchinson, and C. Thauberger. Talking about teaching self-regulated learning: Scaffolding student teachers’ development and use of practices that promote self-regulated learning. *International Journal of Educational Research*, 47(2):97–108, 2008.
- [38] M. Puustinen and L. Pulkkinen. Models of self-regulated learning: A review. *Scandinavian journal of educational research*, 45(3):269–286, 2001.
- [39] J. B. Ramsey. Why do students find statistics so difficult. *Proceedings of the 52th Session of the ISI. Helsinki*, pages 10–18, 1999.
- [40] S. Ritter, J. R. Anderson, K. R. Koedinger, and A. Corbett. Cognitive tutor: Applied research in mathematics education. *Psychonomic bulletin & review*, 14(2):249–255, 2007.
- [41] R. D. Snee. What’s missing in statistical education? *The american statistician*, 47(2):149–154, 1993.
- [42] Y.-h. Tsai, C.-h. Lin, J.-c. Hong, and K.-h. Tai. The effects of metacognition on online learning interest and continuance to learn with moocs. *Computers & Education*, 121:18–29, 2018.
- [43] A. Tversky and D. Kahneman. Availability: A heuristic for judging frequency and probability. *Cognitive psychology*, 5(2):207–232, 1973.
- [44] A. Uzir, D. Gašević, W. Matcha, J. Jovanović, A. Pardo, L.-A. Lim, S. Gentili, et al. Discovering time management strategies in learning processes using process mining techniques. In *European Conference on Technology Enhanced Learning*, pages 555–569. Springer, 2019.
- [45] M. V. Veenman. Learning to self-monitor and self-regulate. In *Handbook of research on learning and instruction*, pages 249–273. Routledge, 2016.
- [46] D. G. Watts. Why is introductory statistics difficult to learn? and what can we do to make it easier? *The American Statistician*, 45(4):290–291, 1991.
- [47] H. Wei, H. Li, M. Xia, Y. Wang, and H. Qu. Predicting student performance in interactive online question pools using mouse interaction features. In *Proceedings of the Tenth International Conference on Learning Analytics & Knowledge*, pages 645–654, 2020.
- [48] B. J. Zimmerman. Self-regulated learning and academic achievement: An overview. *Educational psychologist*, 25(1):3–17, 1990.
- [49] B. J. Zimmerman. Becoming a self-regulated learner: An overview. *Theory Into Practice*, 41(2):64–70, 2002.
- [50] B. J. Zimmerman and M. M. Pons. Development of a structured interview for assessing student use of self-regulated learning strategies. *American educational research journal*, 23(4):614–628, 1986.

Grade Prediction via Prior Grades and Text Mining on Course Descriptions: Course Outlines and Intended Learning Outcomes

Jiawei Li
Nanyang Technological
University
jiawei009@e.ntu.edu.sg

Wei Qiu
Nanyang Technological
University
qiuwei@ntu.edu.sg

S. Supraja
Nanyang Technological
University
supraja.s@ntu.edu.sg

Andy W. H. Khong
Nanyang Technological
University
andykhong@ntu.edu.sg

ABSTRACT

Academic grades in assessments are predicted to determine if a student is at risk of failing a course. Sequential models or graph neural networks that have been employed for grade prediction do not consider relationships between course descriptions. We propose the use of text mining to extract semantic, syntactic, and frequency-based features from course content. In addition, we classify intended learning outcomes according to their higher- or lower-order thinking skills. A learning parameter is then formulated to model the impact of these cognitive levels (that are expected for each course) on student performance. These features are then embedded and represented as graphs. Past academic achievements are then fused with the above features for grade prediction. We validate the performance of the above approach via datasets corresponding to three engineering departments collected from a university. Results obtained highlight that the proposed technique generates meaningful feature representations and outperforms existing methods for grade prediction.

Keywords

Grade prediction, graph networks, course descriptions, semantic similarities, cognitive levels

1. INTRODUCTION

Detecting students at the risk of failing university courses based on predicted grades is essential for administering early intervention strategies. From a regression problem perspective, grades obtained from prior courses in previous semesters are used to predict grades for pilot courses registered in the upcoming semester.

J. Li, S. Supraja, W. Qiu, and A. W. H. Khong. Grade prediction via prior grades and text mining on course descriptions: Course outlines and intended learning outcomes. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 446–453, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853171>

1.1 Related Models for Grade Prediction

Existing techniques for grade prediction using past academic records include conventional regression models such as random forest, support vector machine, and K-nearest neighbor [1, 10, 15] as well as the factorization machine in a collaborative filtering setting [33]. In addition to the use of past examination results, information derived from online click-stream data on learning management systems has been used to augment the prediction capability of a model [25, 26]. More recently, sequential models such as the long short-term memory (LSTM) have been developed to capture the temporal dynamics of past academic performance [12]. While such deep learning models have achieved reasonable success in grade prediction, existing temporal-based approaches do not take the relationships among courses and among students into account. Consideration of these relationships is essential since information pertaining to courses with similar content and students with similar cognitive levels would aid in grade prediction. In addition, the performance trend of an academically-inclined student or a well-performed course in the current semester may continue for the upcoming semesters [23].

Notwithstanding the above, graph neural networks have recently been employed to generate meaningful feature representations which model the transitions of grade distributions between courses across semesters [11]. Similar to social multi-relational networks [14] with nodes representing either students or courses, three graphs—student-course, student-student, and course-course graphs—consisting of edge links computed via grade distribution similarities or correlations have been constructed [21, 23]. Modeling the student-course relations have also been achieved via knowledge graphs to extract course and student embeddings as well as to encode temporal student behavioral data [17]. Pre- or co-requisites between courses have also been considered for grade prediction [27].

Despite adopting multi-dimensional approaches toward analyzing prior course grades to predict student performance [35], existing models assume that the relationship among courses depends solely on the grade distribution; these models do not consider topics covered and the intended

learning outcomes defined by the course instructors. These aspects are important since the process of knowledge acquisition often involves assimilating and discerning information from myriad sources [29], i.e., academic performance has shown to be dependent on prior experience and how the student has understood certain concepts. Moreover, course content that overlap or are highly inter-dependent may influence how well the student can achieve the intended learning outcomes for the upcoming semesters [38]. While course syllabus has recently been used to extract frequency-based features for the determination of course similarities [16], it does not analyze the intended learning outcomes nor capture the relationship between courses holistically. It is also not surprising to expect that students who are less academically inclined often struggle in courses that require higher-order thinking skills. Information pertaining to the thinking skills required for prior courses will, therefore, allow the grade-prediction model to better represent grades achieved from previous semesters.

1.2 Grade Prediction From Curriculum Development Perspective

From a curriculum development perspective, course descriptions comprise topics to be covered and the intended learning outcomes for each course designed by the course instructor [34]. The importance of identifying suitable topics is motivated by an earlier study where first-year university students who had been exposed to fundamental concepts in high school have shown to perform better than those who had not studied similar content before [13]. In today’s context, this highlights the intrinsic (and often intimate) relationships including pre-requisites, recommended literature, and course content that define dependencies between courses. Coupled with the fact that course instructors often adopt the constructivist approach in curriculum design [6], analysis of course content is important for grade prediction.

Apart from course content, outcome-based teaching and learning require course instructors to identify suitable intended learning outcomes and assessments that measure those learning outcomes [4, 30]. In this regard, learning activities with various cognitive complexity levels should be designed and aligned with the learning outcomes constructively throughout the course [2, 5, 9, 32]. Alignment of learning activities can be achieved via the revised Bloom’s Taxonomy with the recollection of information being associated with the lowest-order thinking skill to generating creative outcomes being associated with the highest-order thinking skill [20]. Given that less academically-inclined students often face challenges in higher-order thinking skills [39], it is important to consider the influence of learning outcomes on student performance for grade prediction.

1.3 Contribution of This Work

In this work, we propose a course description-based grade prediction (CODE-GP) model that employs text mining techniques for extracting features associated with (i) course content similarities and (ii) higher- or lower-order thinking skills required for each course. With regards to the first dimension highlighted in Table 1, we propose three types of course similarities extracted from topic outlines and intended learning outcomes found within course descriptions.

Table 1: Overview of text mining approaches in the proposed CODE-GP model

Dimension	Type	Description
Course similarities from course outlines and learning outcomes	Semantic	Contextual closeness of course content
	Syntactic	Grammatical differences across cognitive levels
	Frequency-based	Overlapping works appearance in descriptions
Student similarities based on thinking skills required	Higher-order \mathcal{H}	Verbs corresponding to creative outcomes
	Lower-order \mathcal{L}	Verbs corresponding to recalling concepts

These similarities include semantic [22], syntactic [3], and frequency-based features [36]. The use of these features is in contrast to the use of grade distributions as edge weights for generating similarities [11]. The basis for our proposed architecture is motivated by the need to consider both course outlines and intended learning outcomes, since both the intended learning outcomes and syllabus are important for the development and implementation of teaching programs [28]. In addition, we also consider past performance of each student from the perspective of thinking skills required for each course. In particular, the proposed model employs a document classification approach that tags each course with higher- or lower-order thinking skills according to the revised Bloom’s Taxonomy. A learnable parameter is then used to aggregate the respective grades achieved for both lower- and higher-order thinking skill courses. This allows the proposed model to establish the relationship between the complexity of courses and academic performance.

As shown in Figure 1, we adopt graph neural networks to generate representations of the above text mining features. These features are represented as course- and student-similarity graphs with nodes corresponding to courses and students, respectively. The edge weights for the former are computed based on the proposed three text features. For the latter, past academic grades are aggregated, and the similarity related to Jensen-Shannon Divergence (JSD) is then computed among the grade distributions [23]. These graphs are subsequently embedded and trained using a graph convolutional network (GCN) layer.

In addition and similar to [12], we incorporate temporal information extracted from past examination records for each student across semesters. Grade embeddings, the corresponding student vector, and prior course vectors acquired from the GCN for each semester are then concatenated as a representation vector. This temporal representation serves as the input to LSTM, which exploits the sequential relationships and predicts the grade for a course to be taken in the coming semester.

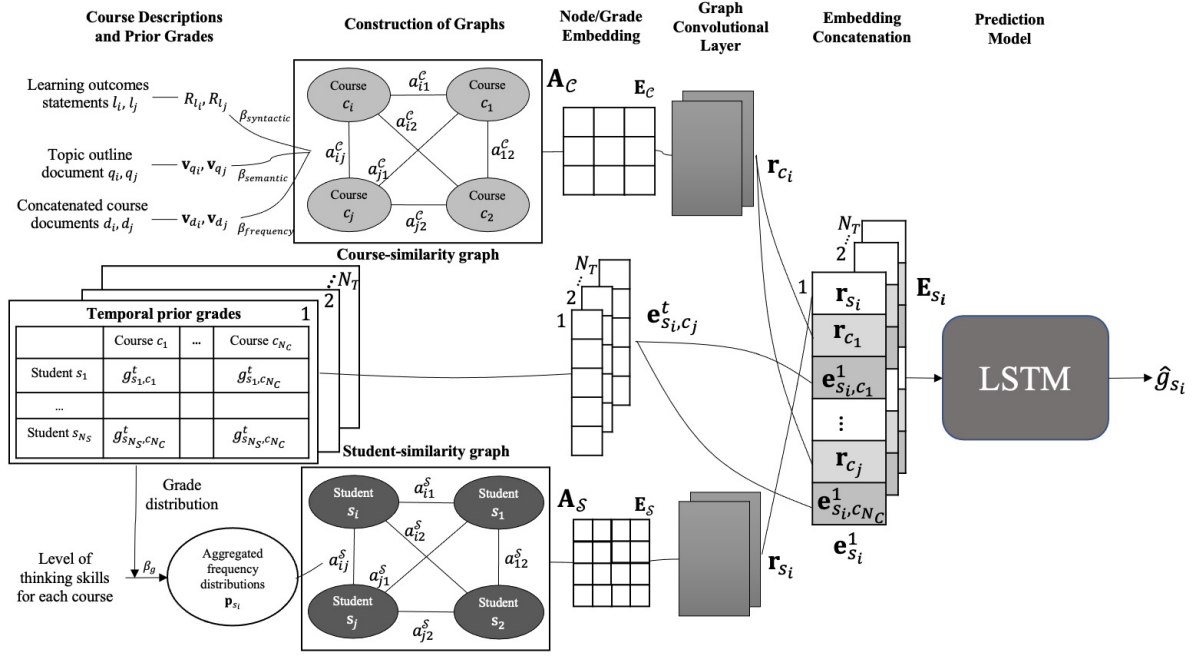


Figure 1: Process flow of the proposed CODE-GP model.

2. THE PROPOSED CODE-GP MODEL

The task of grade prediction involves predicting the grade for student s_i who has registered a pilot course. Given N_C number of prior courses and N_S students, the set of prior courses is defined as $\mathbb{C} = \{c_1, c_2, \dots, c_{N_C}\}$ and the set of students as $\mathbb{S} = \{s_1, s_2, \dots, s_{N_S}\}$. We define \hat{g}_{s_i} as the predicted grade of a given pilot course for the student s_i .

2.1 Construction of Course Similarities Graph Based on Course Descriptions

The CODE-GP model incorporates semantic, syntactic, and frequency-based features extracted from course descriptions that comprise topic outlines and intended learning outcomes. These features are subsequently used for constructing the course-similarity graph. We first pre-process the text by removing symbols, diagrams, equations, numbers, punctuation marks, and stop-words (e.g., “and”, “or”). All remaining characters are set to lower case [32].

Semantic similarity based on word embeddings has been employed to assess student capability for the recommendation of similar courses [24]. In the context of CODE-GP, we first define topic outline as q_i corresponding to course c_i . A topic outline vector \mathbf{v}_{q_i} is then generated from q_i based on the bidirectional encoder representations from transformer (BERT) embeddings [7, 8]. The cosine similarity between q_i and q_j is then computed between two course outlines via

$$\cos(\theta(c_i, c_j)) = \frac{\mathbf{v}_{q_i} \cdot \mathbf{v}_{q_j}}{\|\mathbf{v}_{q_i}\| \|\mathbf{v}_{q_j}\|}. \quad (1)$$

With $0 \leq \cos(\theta(c_i, c_j)) \leq 1$, a value of 1 implies an almost semantically similar pair of courses c_i and c_j .

Syntactic features for CODE-GP comprise phrase types (i.e., regular expressions (regexes)) that are extracted from state-

ments associated with the intended learning outcomes. In this context, we first extract noun- and verb-phrases from the intended learning outcome document l_i corresponding to course c_i . These (multiple) phrases are then associated with their parts-of-speech tags resulting in the set of regexes R_{l_i} [31]. Overlaps between the regex sets are then computed via the Jaccard similarity given by

$$\Phi(c_i, c_j) = \frac{|R_{l_i} \cap R_{l_j}|}{|R_{l_i} \cup R_{l_j}|}, \quad (2)$$

where $0 \leq \Phi(c_i, c_j) \leq 1$. The number of common occurrences is denoted by $|R_{l_i} \cap R_{l_j}|$ while $|R_{l_i} \cup R_{l_j}|$ refers to the total number of regexes. A high Jaccard similarity, therefore, implies a high proportion of similar phrase types occurring between a course pair regardless of whether the topics covered are identical.

The term frequency-inverse document frequency (TF-IDF) determines the uniqueness of a word within a set of documents [37]. To account for word appearance similarity, we include TF-IDF weighting on both the topic outlines q_i and intended learning outcomes l_i for course c_i . These features are extracted from each concatenated course document $d_i = q_i \frown l_i$, where \frown denotes the concatenation of two texts. We then compute the cosine similarity $\Omega(c_i, c_j) \propto \mathbf{v}_{d_i} \cdot \mathbf{v}_{d_j}$ similar to (1) but between bag of words (BoW) vectors \mathbf{v}_{d_i} and \mathbf{v}_{d_j} corresponding to each course document. Here, $\mathbf{v}_{d_i} = [\alpha(w_1, d_i), \alpha(w_2, d_i), \dots, \alpha(w_{N_W}, d_i)]$ with w_k denoting the k th word in document d_i . The BoW vector length is based on the word vocabulary size N_W across the entire corpus. The value of each element corresponding to the TF-IDF weight for word w_k is given by [37]

$$\alpha(w_k, d_i) = \frac{N_{w_k, d_i}}{L(d_i)} \times \log\left(\frac{N_D}{N_{w_i} + 1}\right), \quad (3)$$

where N_{w_k, d_i} is the number of times w_k occurs in d_i , $L(d_i)$ denotes the length of that document, N_D the total number of documents, and N_{w_i} the number of documents in which w_i occurs. The obtained TF-IDF values are subsequently normalized to prevent bias in the term frequency variable due to document length $L(d_i)$.

With nodes of the course-similarity graph denoted by each course $c_i \in \mathbb{C}$, the edge weights are determined via

$$a_{ij}^{\mathbb{C}} \times \left(\beta_{semantic} \times \cos(\theta(c_i, c_j)), \beta_{syntactic} \times \Phi(c_i, c_j), \beta_{frequency} \times \Omega(c_i, c_j) \right), \quad (4)$$

where $\beta_{semantic}$, $\beta_{syntactic}$, and $\beta_{frequency}$ are the trainable weights. Each of the variable $a_{ij}^{\mathbb{C}}$ is used within the adjacency matrix

$$\mathbf{A}_{\mathbb{C}} = \begin{bmatrix} a_{11}^{\mathbb{C}} & \cdots & a_{1N_C}^{\mathbb{C}} \\ \vdots & \ddots & \vdots \\ a_{N_C 1}^{\mathbb{C}} & \cdots & a_{N_C N_C}^{\mathbb{C}} \end{bmatrix} \quad (5)$$

corresponding to the course-similarity graph.

2.2 Temporal Grade Information

Before attempting the pilot course in the current semester, we assume, for each student s_i , availability of prior course grades in \mathbb{C} across semesters $t \in \{1, \dots, N_T\}$, where N_T is the total number of semesters, g_{s_i, c_i}^t denotes the grade that student achieves for c_i in semester t . Hence, the grade vector for student s_i in semester t is given by

$$\mathbf{g}_{s_i}^t = [g_{s_i, c_1}^t, \dots, g_{s_i, c_{N_C}}^t], \quad (6)$$

where N_C is the total number of prior courses across all N_T semesters. It is important to note that for a given semester, only a subset of these N_C prior courses are attempted, i.e., $\mathbf{g}_{s_i}^t$ is not a full vector and null elements will be assigned for courses not attempted during that semester. Across all previous N_T semesters, we acquire the temporal grade information for each student, as shown in Figure 1. Such temporal grade information would be used in two ways—(i) being aggregated according to the thinking skills required for each course and to generate student similarity as will be described in Section 2.3 and (ii) being concatenated with the course and student embeddings as input for LSTM.

2.3 Construction of Student Similarities Graph Based on Cognitive Levels

Construction of the student-similarity graph is based on cognitive levels associated with each course according to Table 1. Each of the prior courses is first categorized as one that requires high-order thinking skills \mathcal{H} or lower-order thinking skills \mathcal{L} . This is achieved by first classifying each course intended learning outcome statement via document classification described in [31] with classes being defined according to Bloom’s Taxonomy. Each course is then tagged as \mathcal{H} (or \mathcal{L}) if more statements are classified as labels associated with high-order (or lower-order) thinking skills.

For each student, we compute the frequency distribution $\mathbf{p}_{s_i}^{\mathbb{C}}$ and $\mathbf{p}_{s_i}^{\mathcal{H}}$ corresponding to courses that require lower- and higher-order thinking skills. This is achieved by first dividing the grade range (1-100) into five bins of twenty-point

Table 2: Details on datasets from three departments

Department	Pilot course index	N_S	N_C	N_T	Number of records
Department 1	c_1	453	16	5	7241
	c_2	645	20	6	11197
Department 2	c_3	575	16	6	9234
	c_4	688	16	6	10977
Department 3	c_5	711	23	7	13616
	c_6	540	17	6	8785

intervals before determining the number of courses (in each \mathcal{H} and \mathcal{L} category) that falls under each bin. Contributions of these two distributions are then learned via

$$\mathbf{p}_{s_i} = \beta_g \times \mathbf{p}_{s_i}^{\mathbb{C}} + (1 - \beta_g) \times \mathbf{p}_{s_i}^{\mathcal{H}}, \quad (7)$$

where β_g is a learnable weight for \mathbf{p}_{s_i} . With the above, student similarities are obtained via the JSD between the grade distribution for each pair of students, i.e.,

$$a_{ij}^{\mathbb{S}} = 1 - JSD(\mathbf{p}_{s_i} || \mathbf{p}_{s_j}). \quad (8)$$

Therefore, a higher $a_{ij}^{\mathbb{S}}$ implies that the two students possess similar higher- or lower-order skills (measured by how they perform in the prior courses). With the student similarity graph shown in Figure 1 comprising students as nodes, the corresponding adjacency matrix $\mathbf{A}_{\mathbb{S}}$ is generated based on $a_{ij}^{\mathbb{S}}$ similar to (5).

2.4 GCN and Embeddings

After constructing the course- and student-similarity graphs, we employ a two-layer GCN to embed each graph. Both course and student nodes are encoded with one-hot vectors to obtain encoded matrices $\mathbf{X}_{\mathbb{C}}$ and $\mathbf{X}_{\mathbb{S}}$. The embedding vector $\mathbf{E}_{\mathbb{C}}$ for the course-similarity graph is generated via

$$\mathbf{E}_{\mathbb{C}} = \mathbf{W}_{\mathbb{C}} \mathbf{X}_{\mathbb{C}} \quad (9)$$

such that the one-hot vectors are represented as dense vectors of lower dimensions. Here, $\mathbf{W}_{\mathbb{C}}$ is the weight matrix. With $\mathbf{E}_{\mathbb{S}}$ being generated similarly, and with $\mathbf{A}_{\mathbb{C}}$ and $\mathbf{A}_{\mathbb{S}}$ derived from Sections 2.1 and 2.3, two GCN layers [18] are then applied to obtain latent representations of all nodes in course-similarity graph \mathbb{C} and student-similarity graph \mathbb{S} . In particular, the $(G + 1)$ th layer for \mathbb{C} is computed via

$$\mathbf{Z}_{\mathbb{C}}^{(G+1)} = \sigma \left(\mathbf{D}_{\mathbb{C}}^{-\frac{1}{2}} \mathbf{A}_{\mathbb{C}} \mathbf{D}_{\mathbb{C}}^{-\frac{1}{2}} \mathbf{Z}_{\mathbb{C}}^{(G)} \mathbf{W}_{\mathbb{C}}^{(G)} \right), \quad (10)$$

where $\mathbf{D}_{\mathbb{C}} = \sum_{c_i} \mathbf{A}_{\mathbb{C}}$ is the degree matrix, $\mathbf{Z}_{\mathbb{C}}^{(0)} = \mathbf{E}_{\mathbb{C}}$, and $\mathbf{W}_{\mathbb{C}}^{(G)}$ is the weight matrix. The output of the GCN for course-similarity graph is denoted as matrix $\mathbf{R}_{\mathbb{C}} = \mathbf{Z}_{\mathbb{C}}^{(2)}$ with each row vector \mathbf{r}_{c_i} being associated with course c_i . The above computation is also applied on student-similarity graph \mathbb{S} to obtain the graph embedding matrix $\mathbf{R}_{\mathbb{S}}$ with each row vector being defined as \mathbf{r}_{s_j} for student s_j .

To generate representations for the prior grades achieved, embedding is applied for each grade. With a one-hot vector representing a unique value of prior grade g_{s_i, c_j}^t , the embedding vector for a student prior grade is learned via

$$\mathbf{e}_{s_i, c_j}^t = \mathbf{W}_G \text{One-hot} \left(g_{s_i, c_j}^t \right), \quad (11)$$

Table 3: Prior courses list for pilot course c_5 from Department 3

EA101	Dynamics	IC102	Physics A	CS108	Computing
EC180	Mathematics 1	EC181	Mathematics 2	EC280	Mathematics A
EA201	Mechanics of Materials	EA202	Theory of Mechanism	EA203	Intro to Thermofluids
EA204	Engineering Materials	EA205	Engineering Graphics	EA206	Engineering Mathematics
EA207	Thermodynamics	EA305	Control Theory	EA306	Fluid Mechanics
EA271	Laboratory Experiments	EA371	Engineering Experiments	EA301	Machine Element Design
EA209	Intro to Electrical Circuits Electronic Devices	EA102	Fundamentals of Engineering Materials	CS103	Introduction to Engineering and Practices
EA302	Solid Mechanics and Vibration	EA304	Mathematical Methods in Engineering		

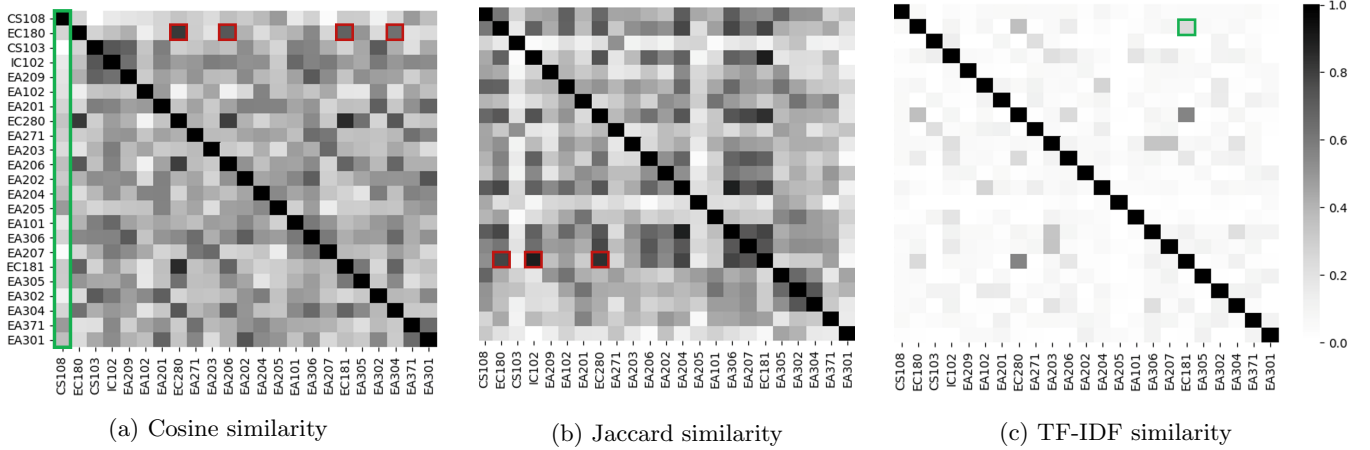


Figure 2: Visualization of three similarities among different prior courses from an engineering department (Department 3) of a university.

where \mathbf{W}_G is the weight matrix. The three embedding vectors from course-similarity graph, student-similarity graph, and temporal grade information are then concatenated for each semester to form a $(l_a + N_C \times (l_b + l_c)) \times 1$ vector

$$\mathbf{e}_{s_i}^t = [\mathbf{r}_{s_i}, \mathbf{r}_{c_1}, \mathbf{e}_{s_i, c_1}^t, \dots, \mathbf{r}_{c_{N_C}}, \mathbf{e}_{s_i, c_{N_C}}^t]^T, \quad (12)$$

where l_a , l_b , and l_c denote the embedding length for \mathbf{r}_{s_i} , \mathbf{r}_{c_j} , and \mathbf{e}_{s_i, c_j}^t , respectively, and T denotes transpose. Each of these vectors are then concatenated to form a feature matrix

$$\mathbf{E}_{s_i} = [\mathbf{e}_{s_i}^1, \dots, \mathbf{e}_{s_i}^{N_T}] \quad (13)$$

of each student s_i for the subsequent prediction model.

2.5 Grade Prediction using LSTM

LSTM models time-series representations and is used to predict the pilot grade based on sequential matrix \mathbf{E}_{s_i} for each student. Through the use of input, output, and forget gate, LSTM aggregates important and permutes less significant representations to achieve prediction of pilot grades in semester $N_T + 1$. LSTM is employed for grade prediction via the hidden state

$$\mathbf{h}_{s_i}^t = \text{LSTM}(\mathbf{e}_{s_i}^t, \mathbf{h}_{s_i}^{t-1}), \quad (14)$$

where $\mathbf{h}_{s_i}^t$ denotes the hidden state for semester t . The predicted grade \hat{g}_{s_i} for student s_i obtained from the last hidden state is then given by

$$\hat{g}_{s_i} = \mathbf{w}_L \cdot \mathbf{h}_{s_i}^{N_T} + b, \quad (15)$$

where \mathbf{w}_L and b are defined, respectively, as the weight vector and bias scalar for the predictor.

3. RESULTS AND DISCUSSION

3.1 Datasets and Implementation Details

Open-source datasets employed for grade prediction do not include course descriptions. We collected data that include both academic records and course descriptions (comprising both course outlines and intended learning outcomes). These are obtained from three engineering departments in a university to evaluate the models. Each dataset is obtained with the student name and identity being hashed by another office (authorized to handle such data) to protect privacy. Table 2 summarizes details for each dataset used. In particular, N_T for each dataset is determined by the maximum number of semesters the students within the cohort take to complete all courses under consideration. The prior course list for each pilot course consists of the core courses corresponding to the department’s curriculum. In addition, N_C and N_S are distinct for each dataset. In our experiments, the training, validation, and testing ratio are set as 6:2:2.

We employed the mean squared error (MSE)

$$\text{MSE} = \frac{1}{N_S} \sum_{i=1}^{N_S} (\hat{g}_{s_i} - g_{s_i})^2 \quad (16)$$

for performance evaluation, where g_{s_i} denotes the actual grade obtained by student s_i for a given pilot course. In terms of hyperparameter selection, course description document embeddings are trained using BERT with a dimension of 768. During GCN training, the dropout rate was set as 0.5, while the Adam optimizer with a learning rate of 0.001 was used. A weight decay parameter was set to 5×10^{-4} to prevent overfitting.

Table 4: Performance evaluation across grade prediction algorithms

Mean Squared Error (MSE)										
Methods	Department 1			Department 2			Department 3			Average
	c_1	c_2	Ave.	c_3	c_4	Ave.	c_5	c_6	Ave.	
LR	0.0360	0.0199	0.0280	0.0262	0.0247	0.0255	0.0264	0.0576	0.0420	0.0318
LSTM [12]	0.0309	0.0210	0.0260	0.0191	0.0259	0.0252	0.0164	0.0377	0.0270	0.0252
GCN [19]	0.0356	0.0214	0.0285	0.0259	0.0251	0.0245	0.0224	0.0276	0.0250	0.0263
Proposed CODE-GP	0.0296	0.0203	0.0250	0.0159	0.0184	0.0172	0.0188	0.0299	0.0244	0.0222

Table 5: Ablation test results

Approach(es)	MSE	MAE
Temporal only (LSTM)	0.0252	0.1288
Graph only (GCN)	0.0263	0.1244
Removal of student-similarity graph	0.0225	0.1223
Removal of course-similarity graph	0.0231	0.1224
Proposed CODE-GP	0.0222	0.1221

3.2 Performance Analysis

We take pilot course c_5 from Department 3 as an example to illustrate the impact of considering the semantic, syntactic, and frequency aspects of words used in course outlines and intended learning outcomes. Three heatmaps with colors depicting the similarity values described in Section 2.2 are provided while details pertaining to prior course information are shown in Table 3.

Figure 2(a) illustrates the semantic cosine similarity where high similarities in terms of the closeness of course content are indicated by the dark shades. It can be seen that the mathematics-based prior course EC180 exhibits high semantic similarity with other prior courses EC280, EA206, EC181, and EA304, which have high mathematical content. On the other hand, computing course CS108 exhibits lower semantic similarity with the most of other non-programming courses. Figure 2(b) highlights how (dis)similar phrase types are between the course outlines and the intended learning outcomes of two prior courses. We note that EC181 exhibits higher Jaccard similarity with courses that require fundamental scientific and mathematical knowledge such as EC180, IC102, and EC280. TF-IDF weighting, on the other hand, indicates the choice and uniqueness of words being used in the course outlines and intended learning outcomes. Figure 2(c) highlights the high variability in words used between the courses being considered—only a few pairs of course outlines and intended learning outcomes exhibit high TF-IDF similarity. In addition, we also note that the similarity between content is irrelevant. This can be observed from the fact that even though EC180 and EC181 are mathematics-related, their frequency-based TF-IDF similarity is relatively low.

We next compare the performance of the proposed CODE-GP model with LSTM based grade-prediction model [12], GCN [19], and the conventional logistic regression (LR) model. While LR and LSTM focus on temporal information and GCN exploits the interrelationship between courses and students, the proposed model considers both aspects. We note from Table 4 that the proposed CODE-GP model achieves the highest grade prediction capability than the LR, LSTM, and GCN. While the proposed model requires higher complexity than these three baseline models, CODE-

GP achieves the lowest mean MSE of 0.0222 (11.9% improvement compared to LSTM), across the three departments as seen in Table 4. These results highlight the importance of course descriptions when constructing student- and course-similarity graphs with time series information. Features extracted from course descriptions enhance the grade prediction capability instead of using only a single modality.

We further performed an ablation test by excluding each input graph/temporal representations. Table 5 summarizes the MSE and mean absolute error (MAE) across all three departments. We note that the use of all three aspects in CODE-GP is vital to provide a holistic perspective for grade prediction. It is interesting to note that grade prediction performance is more sensitive to course-similarity graph (compared to student-similarity graph). This suggests that information derived from course descriptions can assist in grade prediction since performance is closely related to achieving the set of intended learning outcomes depicted in course descriptions. These results also highlight that temporal information and graphs provide complementary features which contribute jointly to the success of grade prediction.

4. CONCLUSIONS

We propose a grade prediction model that considers course descriptions and prior academic results. Text mining techniques determine the edge weights of the course- and student-similarity graphs. A three-pronged model that constitutes the semantic, syntactic, and frequency-based feature extraction methods is formulated for course similarities. Student performance in terms of their achievements in courses associated with low- or high-order thinking skills have also been incorporated to construct the student-similarity graph. The LSTM synthesizes these aspects before performing prediction.

An accurate and just-in-time prediction of performance enables course instructors to administer early interventions. Once the predicted results indicate a tendency of a student in failing a course, student support staff can respond and plan for a personalized intervention strategy for each student. Moreover, early detection of at-risk students can potentially reduce the drop-out rate. Future work may include techniques that incorporate other data modalities such as student demographic or online learning behavior while protecting student privacy.

5. REFERENCES

- [1] M. Adnan, A. Habib, J. Ashraf, S. Mussadiq, A. A. Raza, M. Abid, M. Bashir, and S. U. Khan. Predicting at-risk students at different percentages of course

- length for early intervention using machine learning models. *IEEE Access*, 9:7519–7539, 2020.
- [2] T. Andre. Does answering higher-level questions while reading facilitate productive learning? *Review Edu. Research*, 49:280–318, 1979.
- [3] X. Bai, P. Liu, and Y. Zhang. Investigating typed syntactic dependencies for targeted sentiment classification using graph attention neural network. *IEEE/ACM Trans. Audio Speech Lang. Proc.*, 29:503–514, 2021.
- [4] D. Boud and N. Falchikov. Aligning assessment with long-term learning. *Assessment, Evaluation Higher Edu.*, 31:399–413, 2006.
- [5] S. G. Bull. The role of questions in maintaining attention to textual material. *Review Edu. Research*, 43:83–88, 1973.
- [6] H. Bydžovská. A comparative analysis of techniques for predicting student performance. In *Proc. Int. Conf. Edu. Data Mining (EDM)*, pages 306–311, 2016.
- [7] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proc. Human Lang. Tech.: Annual Conf. North American Chap. (NAACL-HLT)*, pages 4171–4186, 2019.
- [8] M. Fateen and T. Mine. Predicting student performance using teacher observation reports. In *Proc. Int. Conf. Edu. Data Mining (EDM)*, pages 481–486, 2021.
- [9] R. M. Felder and R. Brent. Designing and teaching courses to satisfy the ABET engineering criteria. *J. Eng. Edu.*, 92:7–25, 2003.
- [10] Q. Hu and H. Rangwala. Course-specific Markovian models for grade prediction. In *Proc. Int. Pacific-Asia Conf. Knowledge Discovery Data Mining*, pages 29–41. Springer, 2018.
- [11] Q. Hu and H. Rangwala. Academic performance estimation with attention-based graph convolutional networks. In *Proc. Int. Conf. Educational Data Mining*, pages 69–78, 2019.
- [12] Q. Hu and H. Rangwala. Reliable deep grade prediction with uncertainty estimation. In *Proc. Int. Conf. Learn. Anal. & Knowl.*, pages 76–85, 2019.
- [13] T. Hunt. Overlapping in high school and college again. *J. Edu. Research*, 13(3):197–207, 1926.
- [14] V. N. Ioannidis, A. G. Marques, and G. B. Giannakis. A recurrent graph neural network for multi-relational data. In *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, pages 8157–8161, 2019.
- [15] Z. Iqbal, J. Qadir, A. N. Mian, and F. Kamiran. Machine learning based student grade prediction: A case study. *arXiv*, pages 1–22, 2017.
- [16] W. Jiang and Z. A. Pardos. Evaluating sources of course information and models of representation on a variety of institutional prediction tasks. In *Proc. Int. Conf. Edu. Data Mining (EDM)*, pages 115–125, 2020.
- [17] H. Karimi, T. Derr, J. Huang, and J. Tang. Online academic course performance prediction using relational graph convolutional neural network. In *Proc. Int. Conf. Edu. Data Mining (EDM)*, pages 444–450, 2020.
- [18] T. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *Proc. Int. Conf. Learn. Representations (ICLR)*, pages 1–14, 2017.
- [19] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *Proc. Int. Conf. Learn. Representations*, pages 76–85, 2017.
- [20] D. Krathwohl. A revision of Bloom’s Taxonomy: An overview. *Theory into Practice*, 41:212–218, 2002.
- [21] D. D. Leeds, T. Zhang, and G. M. Weiss. Mining course groupings using academic performance. In *Proc. Int. Conf. Edu. Data Mining (EDM)*, pages 1–5, 2021.
- [22] X. Liu, X. You, X. Zhang, J. Wu, and P. Lv. Tensor graph convolutional networks for text classification. In *Proc. AAAI Conf. Artificial Intell.*, pages 8409–8416, 2020.
- [23] X. Lu, Y. Zhu, Y. Xu, and J. Yu. Learning from multiple dynamic graphs of student and course interactions for student grade predictions. *Neurocomputing*, 431:23–33, 2021.
- [24] H. Ma, X. Wang, J. Hou, and Y. Lu. Course recommendation based on semantic similarity analysis. In *Proc. IEEE Int. Conf. Control Sci. Syst. Engg.*, pages 638–641, 2017.
- [25] K. H. R. Ng, S. Tatinati, and A. W. H. Khong. Online education evaluation for signal processing course through student learning pathways. In *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Process.*, pages 6458–6462, 2018.
- [26] K. H. R. Ng, S. Tatinati, and A. W. H. Khong. Grade prediction from multi-valued click-stream traces via Bayesian-regularized deep neural networks. *IEEE Trans. Signal Process.*, 69:1477–1491, 2021.
- [27] Z. Ren, X. Ning, A. S. Lan, and H. Rangwala. Grade prediction based on cumulative knowledge and co-taken courses. In *Proc. Int. Conf. Educational Data Mining*, pages 158–167, 2019.
- [28] J. C. Richards. Curriculum approaches in language teaching: Forward, central, and backward design. *RELJ*, 44(1):5–33, 2013.
- [29] S. H. Seyyedrezaie and G. Barani. Constructivism and curriculum development. *J. Humanities Insights*, 1(3):119–124, 2017.
- [30] S. Supraja, K. Hartman, S. Tatinati, and A. W. H. Khong. Toward the automatic labeling of course questions for ensuring their alignment with learning outcomes. In *Proc. 10th Int. Conf. Educational Data Mining (EDM)*, pages 56–63, 2017.
- [31] S. Supraja, A. W. H. Khong, and S. Tatinati. Regularized phrase-based topic model for automatic question classification with domain-agnostic class labels. *IEEE/ACM Trans. Audio Speech Lang. Proc.*, 29:3604–3616, 2021.
- [32] S. Supraja, S. Tatinati, K. Hartman, and A. W. H. Khong. Automatically linking digital signal processing assessment questions to key engineering learning outcomes. In *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, pages 6996–7000, 2018.
- [33] M. Sweeney, J. Lester, and H. Rangwala. Next-term student grade prediction. In *Proc. IEEE Int. Conf. Big Data*, pages 970–975, 2015.
- [34] R. Tang and W. Sae-Lim. Data science programs in U.S. higher education: An exploratory content

- analysis of program description, curriculum structure, and course focus. *Edu. Info.*, 32(3):269–290, 2016.
- [35] J. Valençon and M. Coates. Multiple-graph recurrent graph convolutional neural network architectures for predicting disease outcomes. In *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, pages 3157–3161, 2019.
- [36] P. Wei, J. Zhao, and W. Mao. A graph-to-sequence learning framework for summarizing opinionated texts. *IEEE/ACM Trans. Audio Speech Lang. Proc.*, 29:1650–1660, 2021.
- [37] A. A. Yahya, A. Osman, A. Taleb, and A. A. Alattab. Analyzing the cognitive level of classroom questions using machine learning techniques. In *Proc. 9th Int. Conf. Cognitive Sci.*, pages 587–595, 2013.
- [38] Y. Zhang, R. An, S. Liu, J. Cui, and X. Shang. Predicting and understanding student learning performance using multi-source sparse attention convolutional neural networks. *IEEE Transactions on Big Data*, 2021.
- [39] A. Zohar and Y. J. Dori. Higher order thinking skills and low-achieving students: Are they mutually exclusive? *J. Learn. Sci.*, 12(2):145–181, 2003.

From {Solution} Synthesis to {Student Attempt} Synthesis for Block-Based Visual Programming Tasks*

Adish Singla
MPI-SWS
adishs@mpi-sws.org

Nikitas Theodoropoulos
MPI-SWS
ntheodor@mpi-sws.org

ABSTRACT

Block-based visual programming environments are increasingly used to introduce computing concepts to beginners. Given that programming tasks are open-ended and conceptual, novice students often struggle when learning in these environments. AI-driven programming tutors hold great promise in automatically assisting struggling students, and need several components to realize this potential. We investigate the crucial component of student modeling, in particular, the ability to automatically infer students’ misconceptions for predicting (synthesizing) their behavior. We introduce a novel benchmark, STUDENTSYN, centered around the following challenge: For a given student, synthesize the student’s attempt on a new target task after observing the student’s attempt on a fixed reference task. This challenge is akin to that of program synthesis; however, instead of synthesizing a {solution} (i.e., program an expert would write), the goal here is to synthesize a {student attempt} (i.e., program that a given student would write). We first show that human experts (TUTORSS) can achieve high performance on the benchmark, whereas simple baselines perform poorly. Then, we develop two neuro/symbolic techniques (NEURSS and SYMSS) in a quest to close this gap with TUTORSS.

Keywords

block-based visual programming, programming education, program synthesis, neuro-symbolic AI, student modeling

1. INTRODUCTION

The emergence of block-based visual programming platforms has made coding more accessible and appealing to beginners. Block-based programming uses “code blocks” that reduce the burden of syntax and introduces concepts in an interactive way. Led by initiatives like *Hour of Code* by Code.org [10, 8] and the popularity of languages like *Scratch* [41], block-based programming has become integral to introductory CS

*Correspondence to: Adish Singla <adishs@mpi-sws.org>; Authors listed alphabetically.

A. Singla and N. Theodoropoulos. From solution synthesis to student attempt synthesis for block-based visual programming tasks. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 454–461, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853165>

education. Considering the *Hour of Code* initiative alone, over one billion hours of programming activity has been spent in learning to solve tasks in such environments [8].

Programming tasks on these platforms are conceptual and open-ended, and require multi-step deductive reasoning to solve. Given these aspects, novices often struggle when learning to solve these tasks. The difficulties faced by novice students become evident by looking at the trajectory of students’ attempts who are struggling to solve a given task. For instance, in a dataset released by Code.org [10, 8, 35], even for simple tasks where solutions require only 5 code blocks (see Figure 1a), students submitted over 50,000 unique attempts with some exceeding a size of 50 code blocks.

AI-driven programming tutors have the potential to support these struggling students by providing personalized assistance, e.g., feedback as hints or curriculum design [37]. To effectively assist struggling students, AI-driven systems need several components, a crucial one being *student modeling*. In particular, we need models that can automatically infer a student’s knowledge from limited interactions and then predict the student’s behavior on new tasks. However, student modeling in block-based visual programming environments can be quite challenging because of the following: (i) programming tasks are conceptual with no well-defined skill-set or problem-solving strategy for mastery [23]; (ii) there could be a huge variability in students’ attempts for a task [52]; (iii) the objective of predicting a given student’s behavior on new tasks is not limited to coarse-grained success/failure indicators (e.g., [50])—ideally, we should be able to do fine-grained synthesis of attempts for the student.

Beyond the above-mentioned challenges, there are two critical issues arising from limited resources and data scarcity for a given domain. First, while the space of tasks that could be designed for personalized curriculum is intractably large [1], the publicly available datasets of real-world students’ attempts are limited; e.g., the *Hour of Code: Maze Challenge* domain has datasets for only two tasks [35]. Second, when a deployed system is interacting with a new student, there is limited prior information [15], and the system would have to infer the student’s knowledge by observing behavior on a few reference tasks, e.g., through a quiz [21]. These two issues limit the applicability of state-of-the-art techniques that rely on large-scale datasets across tasks or personalized data per student (e.g., [50, 28, 29, 36])—we need next-generation student modeling techniques that can operate under data

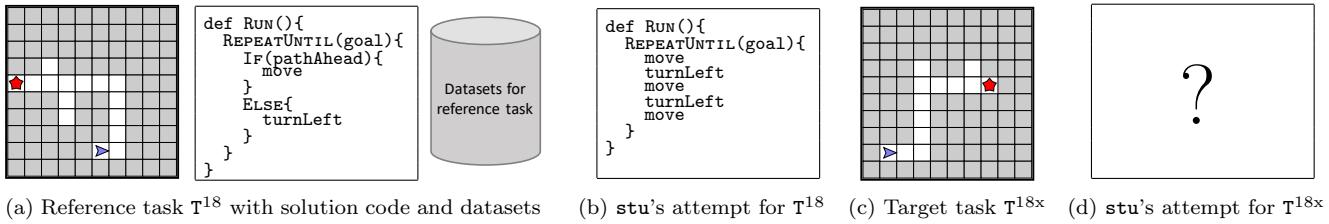


Figure 1: Illustration of our problem setup and objective for the task Maze#18 in the *Hour of Code: Maze* [9] by Code.org [8]. As explained in Section 2.2, we consider three distinct phases in our problem setup to provide a conceptual separation in terms of information and computation available to a system. (a) In the first phase, we are given a reference task T^{18} along with its solution code $C_{T^{18}}^*$ and data resources (e.g., a real-world dataset of different students’ attempts); reference tasks are fixed and the system can use any computation a priori. (b) In the second phase, the system interacts with a student, namely *stu*, who attempts the reference task T^{18} and submits a code, denoted as $C_{T^{18}}^{\text{stu}}$. (c, d) In the third phase, the system seeks to synthesize the student *stu*’s behavior on a target task T^{18x} , i.e., a program that *stu* would write if the system would assign T^{18x} to the student. Importantly, the target task T^{18x} is not available a priori and this synthesis process would be done in real-time.

scarcity and limited observability. To this end, this paper focuses on the following question: *For a given student, can we synthesize the student’s attempt on a new target task after observing the student’s attempt on a fixed reference task?*

1.1 Our Approach and Contributions

Figure 1 illustrates this synthesis question for a scenario in the *Hour of Code: Maze Challenge* [9] by Code.org [8]. This question is akin to that of program synthesis [20]; however, instead of synthesizing a *{solution}* (i.e., program an expert would write), the goal here is to synthesize a *{student attempt}* (i.e., program that a given student would write). This goal of synthesizing student attempts, and not just solutions, requires going beyond state-of-the-art program synthesis techniques [3, 4, 25]; crucially, we also need to define appropriate metrics to quantitatively measure the performance of different techniques. Our main contributions are:

- (1) We formalize the problem of synthesizing a student’s attempt on target tasks after observing the student’s behavior on a fixed reference task. We introduce a novel benchmark, STUDENTSYN, centered around the above synthesis question, along with generative/discriminative performance measures for evaluation.
- (2) We showcase that human experts (TUTORSS) can achieve high performance on STUDENTSYN, whereas simple baselines perform poorly.
- (3) We develop two techniques inspired by neural (NEURSS) and symbolic (SYMSS) methods, in a quest to close the gap with human experts (TUTORSS).

We provide additional details and results in the longer version of the paper [47]. We will also publicly release the benchmark and implementations to facilitate future research.

1.2 Related Work

Student modeling. For close-ended domains like vocabulary learning ([42, 36, 22]) and Algebra problems ([12, 40, 43]), the skills or knowledge components for mastery are typically well-defined and we can use *Knowledge Tracing* techniques to model a student’s knowledge state over time [11, 33]. These modeling techniques, in turn, allow us to provide feedback, predict solution strategies, or infer/quiz a student’s knowledge state [40, 21, 43]. Open-ended domains pose unique challenges to directly apply these techniques

(see [23]); however, there has been some progress in this direction. In recent works [28, 29], models have been proposed to predict human behavior in chess for specific skill levels and to recognize the behavior of individual players. Along these lines, [7] introduced methods to perform early prediction of struggling students in open-ended interactive simulations. There has also been work on student modeling for block-based programming, e.g., clustering-based methods for misconception discovery [18, 44], and deep learning methods to represent knowledge and predict performance [50].

AI-driven systems for programming education. There has been a surge of interest in developing AI-driven systems for programming education, and in particular, for block-based programming domains [37, 38, 51]. Existing works have studied various aspects of intelligent feedback, for instance, providing next-step hints when a student is stuck [35, 53, 31, 15], giving data-driven feedback about a student’s misconceptions [45, 34, 39, 52], or generating/recommending new tasks [2, 1, 19]. Depending on the availability of datasets and resources, different techniques are employed: using historical datasets to learn code embeddings [34, 31], using reinforcement learning in zero-shot setting [15, 46], bootstrapping from a small set of expert annotations [34], or using expert grammars to generate synthetic training data [52].

Neuro-symbolic program synthesis. Our approach is related to program synthesis, i.e., automatically constructing programs that satisfy a given specification [20]. The usage of deep learning models for program synthesis has resulted in significant progress in a variety of domains including string transformations [16, 14, 32], block-based visual programming [3, 4, 13, 48], and competitive programming [25]. Program synthesis has also been used to learn compositional symbolic rules and mimic abstract human learning [30, 17].

2. PROBLEM SETUP

Next, we introduce definitions and formalize our objective.

2.1 Preliminaries

The space of tasks. We define the space of tasks as \mathbb{T} ; in this paper, \mathbb{T} is inspired by the popular *Hour of Code: Maze Challenge* [9] from Code.org [8]; see Figure 1a. We define a task $T \in \mathbb{T}$ as a tuple $(T_{\text{vis}}, T_{\text{store}}, T_{\text{size}})$, where T_{vis} denotes a visual puzzle, T_{store} the available block types, and T_{size} the maximum number of blocks allowed in the solution

code. The task T in Figure 1a corresponds to Maze#18 in the *Hour of Code: Maze Challenge* [9], and has been studied in a number of prior works [35, 15, 1].

The space of codes. We define the space of all possible codes as \mathbb{C} and represent them using a *Domain Specific Language* (DSL) [20]. In particular, for codes relevant to tasks considered in this paper, we use a DSL from [1]. A code $C \in \mathbb{C}$ has the following attributes: $\mathbb{C}_{\text{blocks}}$ is the set of types of code blocks used in C , \mathbb{C}_{size} is the number of code blocks used, and $\mathbb{C}_{\text{depth}}$ is the depth of the *Abstract Syntax Tree* of C .

Solution code and student attempt. For a given task T , a *solution code* $C_T^* \in \mathbb{C}$ should solve the visual puzzle; additionally, it can only use the allowed types of code blocks (i.e., $\mathbb{C}_{\text{blocks}} \subseteq \mathbb{T}_{\text{store}}$) and should be within the specified size threshold (i.e., $\mathbb{C}_{\text{size}} \leq \mathbb{T}_{\text{size}}$). We note that a task $T \in \mathbb{T}$ may have multiple solution codes; in this paper, we typically refer to a single solution code that is provided as input. A *student attempt* for a task T refers to a code that is being written by a student (including incorrect or partial codes). A student attempt could be any code $C \in \mathbb{C}$ as long as it uses the set of available types of code blocks (i.e., $\mathbb{C}_{\text{blocks}} \subseteq \mathbb{T}_{\text{store}}$).

2.2 Objective

Distinct phases. To formalize our objective, we introduce three distinct phases in our problem setup that provide a conceptual separation in terms of information and computation available to a system. More concretely, we have:

- (1) Reference task T^{ref} : We are given a reference task T^{ref} for which we have real-world datasets of different students’ attempts as well as access to other data resources. Reference tasks are fixed and the system can use any computation a priori (e.g., compute code embeddings).
- (2) Student stu attempts T^{ref} : The system interacts with a student, namely stu , who attempts the reference task T^{ref} and submits a code, denoted as $C_{T^{\text{ref}}}^{\text{stu}}$. At the end of this phase, the system has observed stu ’s behavior on T^{ref} and we denote this observation by the tuple $(T^{\text{ref}}, C_{T^{\text{ref}}}^{\text{stu}})$.¹
- (3) Target task T^{tar} : The system seeks to synthesize the student stu ’s behavior on a target task T^{tar} . Importantly, the target task T^{tar} is not available a priori and this synthesis process would be done in real-time, possibly with constrained computational resources. Furthermore, the system may have to synthesize the stu ’s behavior on a large number of different target tasks from the space \mathbb{T} (e.g., to personalize the next task in a curriculum).²

Granularity level of our objective. There are several different granularity levels at which we can predict the student stu ’s behavior for T^{tar} , including: (a) a coarse-level binary prediction of whether stu will successfully solve T^{tar} , (b) a medium-level prediction about stu ’s behavior w.r.t. to a predefined feature set (e.g., labelled misconceptions); (c) a fine-level prediction in terms of synthesizing $C_{T^{\text{tar}}}^{\text{stu}}$, i.e., a program that stu would write if the system would assign

¹In practice, the system might have more information, e.g., the whole trajectory of edits leading to $C_{T^{\text{ref}}}^{\text{stu}}$.

²Even though the *Hour of Code: Maze Challenge* [9] has only 20 tasks, the space \mathbb{T} is intractably large and new tasks can be generated, e.g., for providing feedback [1].

T^{tar} to the student. In this work, we focus on this fine-level, arguably also the most challenging, synthesis objective.

Performance evaluation. So far, we have concretized the synthesis objective; however, there is still a question of how to quantitatively measure the performance of a technique set out to achieve this objective. The key challenge stems from the open-ended and conceptual nature of programming tasks. Even for seemingly simple tasks such as in Figure 1a, the students’ attempts can be highly diverse, thereby making it difficult to detect a student’s misconceptions from observed behaviors; moreover, the space of misconceptions itself is not clearly understood. To this end, we begin by designing a benchmark to quantitatively measure the performance of different techniques w.r.t. our objective.

3. BENCHMARK

In this section, we introduce our benchmark, STUDENTSYN.

3.1 STUDENTSYN: Data Curation

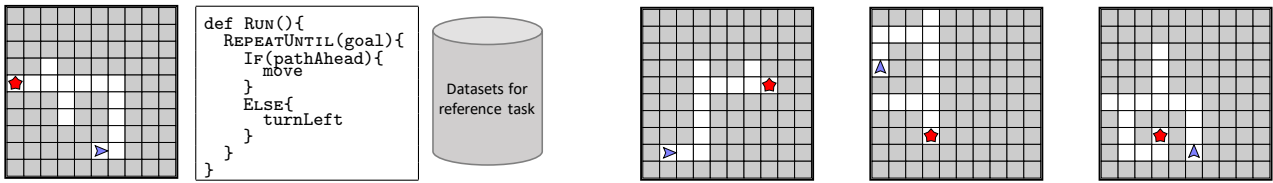
We begin by curating a synthetic dataset for the benchmark, designed to capture different scenarios of the three distinct phases mentioned in Section 2.2. In particular, each scenario corresponds to a 4-tuple $(T^{\text{ref}}, C_{T^{\text{ref}}}^{\text{stu}}, T^{\text{tar}}, C_{T^{\text{tar}}}^{\text{stu}})$, where $C_{T^{\text{ref}}}^{\text{stu}}$ (observed by the system) and $C_{T^{\text{tar}}}^{\text{stu}}$ (to be synthesized by the system) correspond to a student stu ’s attempts.

Reference and target tasks. We select two reference tasks for this benchmark, namely T^4 and T^{18} —they correspond to Maze#4 and Maze#18 in the *Hour of Code: Maze Challenge* [9]. These tasks have been studied in a number of prior works [35, 15, 1] because of the availability of large-scale datasets of students’ attempts. For each reference task, we manually create three target tasks—Figure 2b illustrates target tasks for T^{18} ; the target tasks for T^4 can be found in the longer version of the paper [47]. These target tasks are similar to the corresponding reference task in a sense that the set of available block types is same and the nesting structure of programming constructs in solution codes is same.

Types of students’ behaviors and students’ attempts. For a given reference-target task pair $(T^{\text{ref}}, T^{\text{tar}})$, next we seek to simulate a student stu to create stu ’s attempts $C_{T^{\text{ref}}}^{\text{stu}}$ and $C_{T^{\text{tar}}}^{\text{stu}}$. We begin by identifying a set of salient students’ behaviors and misconceptions for reference tasks T^4 and T^{18} based on students’ attempts observed in the real-world dataset of [35]. In this benchmark, we select 6 types of students’ behaviors for each reference task—Figure 2c highlights the 6 selected types for T^{18} ; the 6 selected types for T^4 can be found in the longer version of the paper [47].³ For a given pair $(T^{\text{ref}}, T^{\text{tar}})$, we first simulate a student stu by associating this student to one of the 6 types, and then manually create stu ’s attempts $C_{T^{\text{ref}}}^{\text{stu}}$ and $C_{T^{\text{tar}}}^{\text{stu}}$. For a given scenario $(T^{\text{ref}}, C_{T^{\text{ref}}}^{\text{stu}}, T^{\text{tar}}, C_{T^{\text{tar}}}^{\text{stu}})$, the attempt $C_{T^{\text{tar}}}^{\text{stu}}$ is not observed and serves as a *ground truth* for evaluation purposes; henceforth, we interchangeably write a scenario as $(T^{\text{ref}}, C_{T^{\text{ref}}}^{\text{stu}}, T^{\text{tar}}, ?)$.

Total scenarios. We create 72 scenarios $(T^{\text{ref}}, C_{T^{\text{ref}}}^{\text{stu}}, T^{\text{tar}}, C_{T^{\text{tar}}}^{\text{stu}})$ in the benchmark corresponding to (i) 2 reference tasks, (ii) 3 target tasks per reference task, (iii) 6 types of students’ behaviors per reference task, and (iv) 2 students per type.

³We note that, in real-world settings, the types of students’ behaviors and their attempts have a much larger variability and complexities with a long-tail distribution.



(a) Reference task T^{18} with solution code and datasets

(b) Three target tasks for T^{18} : T^{18x} , T^{18y} , and T^{18z}

(c) Example codes (i)–(vi) corresponding to six types of students’ behaviors when attempting T^{18} , each capturing different misconceptions

Figure 2: Illustration of the key elements of the STUDENTSYN benchmark for the reference task T^{18} shown in (a)—same as in Figure 1a. (b) Shows three target tasks associated with T^{18} ; these target tasks are similar to T^{18} in a sense that the set of available block types is same as T_{store}^{18} and the nesting structure of programming constructs in solution codes is same as in $C_{T^{18}}^*$. (c) Shows example codes corresponding to six types of students’ behaviors when attempting T^{18} , each capturing a different misconception as follows: (i) confusing left/right directions when turning or checking conditionals, (ii) following one of the wrong path segments, (iii) misunderstanding of IFELSE structure functionality and writing the same blocks in both the execution branches, (iv) ignoring the IFELSE structure when solving the task, (v) ignoring the WHILE structure when solving the task, (vi) attempting to solve the task by using only the basic action blocks in {turnLeft, turnRight, move}.

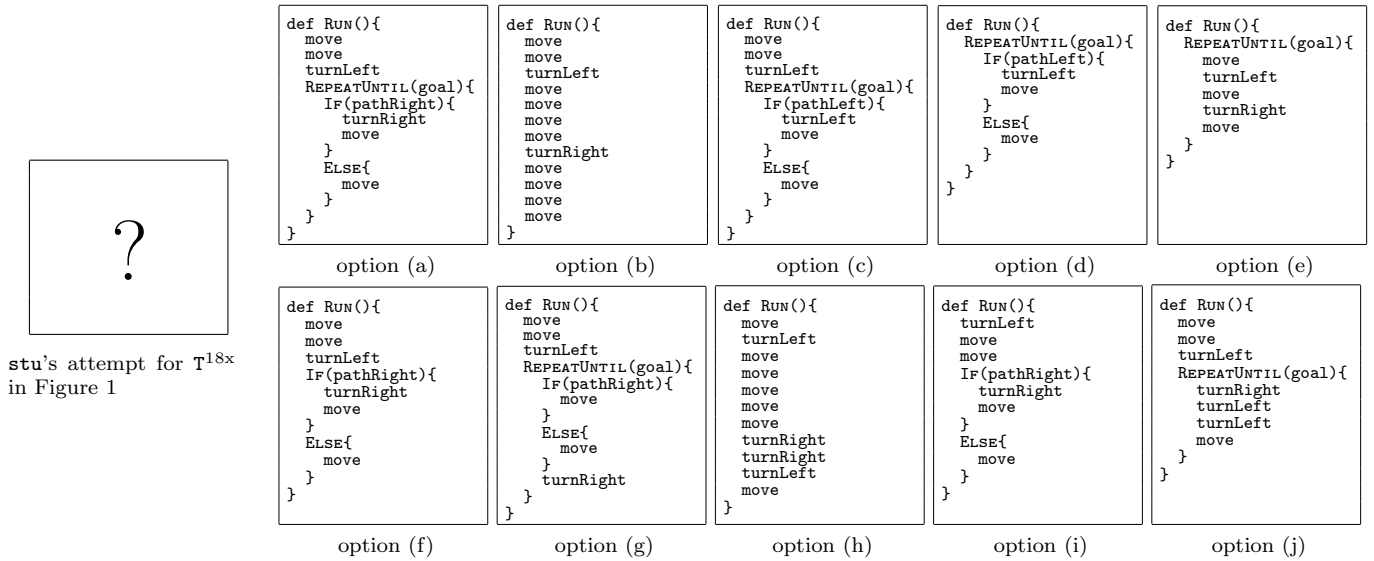


Figure 3: Illustration of the generative and discriminative objectives in the STUDENTSYN benchmark for the scenario shown in Figure 1. For the generative objective, the goal is to synthesize the student *stu*’s behavior on the target task T^{18x} , i.e., a program that *stu* would write if the system would assign T^{18x} to the student. For the discriminative objective, the goal is to choose one of the ten codes, shown as options (a)–(j), that corresponds to the student *stu*’s attempt. For each scenario, ten options are created systematically as discussed in Section 3.2; in this illustration, option (a) corresponds to the solution code $C_{T^{18x}}^*$ for the target task and option (e) corresponds to the student *stu*’s attempt as designed in the benchmark.

3.2 STUDENTSYN: Performance Measures

We introduce two performance measures to capture our synthesis objective. Our first measure, namely *generative performance*, is to directly capture the quality of fine-level synthesis of the student *stu*’s attempt—this measure requires a human-in-the-loop evaluation. To further automate the evaluation process, we then introduce a second performance measure, namely *discriminative performance*.

Generative performance. As a generative performance measure, we introduce a 4-point *Likert scale* to evaluate the quality of synthesizing *stu*’s attempt $C_{T^{tar}}^{stu}$ for a scenario $(T^{ref}, C_{T^{ref}}^{stu}, T^{tar}, ?)$. The scale is designed to assign scores based on two factors: (a) whether the elements of the student’s behavior observed in $C_{T^{ref}}^{stu}$ are present, (b) whether the elements of the target task T^{tar} (e.g., parts of its solution) are present. More concretely, the scores are assigned as

follows (with higher scores being better): (i) Score 1 means the technique does not have synthesis capability; (ii) Score 2 means the synthesis fails to capture the elements of $\mathbf{C}_{\text{Tref}}^{\text{stu}}$ and \mathbf{T}^{tar} ; (iii) Score 3 means the synthesis captures the elements only of $\mathbf{C}_{\text{Tref}}^{\text{stu}}$ or of \mathbf{T}^{tar} , but not both; (iv) Score 4 means the synthesis captures the elements of both $\mathbf{C}_{\text{Tref}}^{\text{stu}}$ and \mathbf{T}^{tar} .

Discriminative performance. As the generative performance measure requires human-in-the-loop evaluation, we also introduce a discriminative performance measure based on the prediction accuracy of choosing the student attempt from a set. More concretely, given a scenario $(\mathbf{T}^{\text{ref}}, \mathbf{C}_{\text{Tref}}^{\text{stu}}, \mathbf{T}^{\text{tar}}, ?)$, the discriminative objective is to choose $\mathbf{C}_{\text{Ttar}}^{\text{stu}}$ from ten candidate codes; see Figure 3. These ten options are created automatically in a systematic way and include: (a) the *ground-truth* $\mathbf{C}_{\text{Ttar}}^{\text{stu}}$, (b) the solution code $\mathbf{C}_{\text{Ttar}}^*$, (c) five codes $\mathbf{C}_{\text{Ttar}}^{\text{stu}'}$ from the benchmark associated with other students stu' whose behavior type is different from stu , and (iv) three *randomly* constructed codes obtained by editing $\mathbf{C}_{\text{Ttar}}^*$.

4. METHODOLOGY

In this section, we design different techniques for the benchmark STUDENTSYN. First, we consider a few simple baselines for the discriminative-only objective (RANDD, EDITD, EDITEMBD). Next, we develop our two main techniques inspired by neural/symbolic methods (NEURSS, SYMSS). Finally, we propose performance evaluation of human experts (TUTORSS). Table 1 illustrates how these techniques differ in required inputs and domain knowledge. Below, we provide a brief overview of these techniques; we refer the reader to the longer version of the paper for full details [47].

Simple baselines. As a starting point, we consider simple baselines for the discriminative-only objective; they do not have synthesis capability. Our first baseline RANDD simply chooses a code from the 10 options at random. Our next two baselines, EDITD and EDITEMBD, are defined through a distance function $D_{\text{Tref}}(\mathbf{C}, \mathbf{C}')$ that quantifies a notion of distance between any two codes \mathbf{C}, \mathbf{C}' for a fixed reference task. For a scenario $(\mathbf{T}^{\text{ref}}, \mathbf{C}_{\text{Tref}}^{\text{stu}}, \mathbf{T}^{\text{tar}}, ?)$ and ten option codes, these baselines select the code \mathbf{C} that minimizes $D_{\text{Tref}}(\mathbf{C}, \mathbf{C}_{\text{Tref}}^{\text{stu}})$. EDITD uses a tree-edit distance between *Abstract Syntax Trees* as the distance function, denoted as $D_{\text{Tref}}^{\text{edit}}$. EDITEMBD extends EDITD by considering a distance function that combines $D_{\text{Tref}}^{\text{edit}}$ and a code-embedding based distance function $D_{\text{Tref}}^{\text{emb}}$; in this paper, we trained code embeddings with the methodology of [15] using a real-world dataset of student attempts on \mathbf{T}^{ref} . EDITEMBD then uses a distance function as a convex combination $(\alpha \cdot D_{\text{Tref}}^{\text{edit}}(\mathbf{C}, \mathbf{C}') + (1 - \alpha) \cdot D_{\text{Tref}}^{\text{emb}}(\mathbf{C}, \mathbf{C}'))$ where α is optimized for each reference task separately.

Neural synthesizer NEURSS. Next, we develop our technique, NEURSS (*Neural Program Synthesis for STUDENTSYN*), inspired by recent advances in neural program synthesis [3, 4]. A neural synthesizer model takes as input a visual task \mathbf{T} , and then sequentially synthesizes a code \mathbf{C} by using programming tokens in $\mathbf{T}_{\text{store}}$. However, our goal is not simply to synthesize a solution code for the input task \mathbf{T} as considered in [3, 4]; instead, we want to synthesize attempts of a given student that the system is interacting with at real-time/deployment. To achieve this goal, NEURSS operates in three stages, where each stage is in line with a phase of our objective described in Section 2.2. At a high-level, the three

stages of NEURSS are as follows: (i) In Stage-1, we are given a reference task and its solution $(\mathbf{T}^{\text{ref}}, \mathbf{C}_{\text{Tref}}^*)$, and train a neural synthesizer model that can synthesize solutions for any task similar to \mathbf{T}^{ref} ; (ii) In Stage-2, the system observes the student stu 's attempt $\mathbf{C}_{\text{Tref}}^{\text{stu}}$ and initiates *continual training* of the neural synthesizer model from Stage-1 in real-time; (iii) In Stage-3, the system considers a target task \mathbf{T}^{tar} and uses the model from Stage-2 to synthesize $\mathbf{C}_{\text{Ttar}}^{\text{stu}}$.

Symbolic synthesizer SYMSS. As we will see in experiments, NEURSS significantly outperforms the simple baselines introduced earlier; yet, there is a substantial gap in the performance of NEURSS and human experts (i.e., TUTORSS). An important question that we seek to resolve is how much of this performance gap can be reduced by leveraging domain knowledge such as how students with different behaviors (misconceptions) write codes. To this end, we develop our technique, SYMSS (*Symbolic Program Synthesis for STUDENTSYN*), inspired by recent advances in using symbolic methods for program synthesis [24, 52, 1, 26]. Similar in spirit to NEURSS, SYMSS operates in three stages as follows: (i) In Stage-1, we are given $(\mathbf{T}^{\text{ref}}, \mathbf{C}_{\text{Tref}}^*)$, and design a symbolic synthesizer model using *Probabilistic Context Free Grammars* (PCFG) to encode how students of different behavior types \mathcal{M} write codes for any task similar to \mathbf{T}^{ref} [5, 27, 52]; (ii) In Stage-2, the system observes the student stu 's attempt $\mathbf{C}_{\text{Tref}}^{\text{stu}}$ and makes a prediction about the behavior type $\mathbf{M}^{\text{stu}} \in \mathcal{M}$; (iii) In Stage-3, the system considers a target task \mathbf{T}^{tar} and uses the model from Stage-1 to synthesize $\mathbf{C}_{\text{Ttar}}^{\text{stu}}$ based on the inferred \mathbf{M}^{stu} .

Human experts. Finally, we propose an evaluation of human experts' performance on the benchmark STUDENTSYN, and refer to this evaluation technique as TUTORSS. These evaluations are done through a web platform where an expert would provide a generative or discriminative response to a given scenario $(\mathbf{T}^{\text{ref}}, \mathbf{C}_{\text{Tref}}^{\text{stu}}, \mathbf{T}^{\text{tar}}, ?)$. In our work, TUTORSS involved participation of three independent experts for the evaluation—these experts have had experience in block-based programming and tutoring. We first carry out generative evaluations where an expert has to write the student attempt code; afterwards, we carry out discriminative evaluations where an expert would choose one of the options.

5. EXPERIMENTAL EVALUATION

In this section, we evaluate the performance of different techniques discussed in Section 4. Our results are summarized in Table 1 and Figure 4. Below, we provide a brief overview of the evaluation procedures and results; we refer the reader to the longer version of the paper for full details [47].

Generative performance. As discussed in Section 3.2, we evaluate the generative performance of a technique in the following steps: (a) a scenario $(\mathbf{T}^{\text{ref}}, \mathbf{C}_{\text{Tref}}^{\text{stu}}, \mathbf{T}^{\text{tar}}, ?)$ is picked; (b) the technique synthesizes stu 's attempt; (c) the generated code is scored on the 4-point *Likert scale*. The scoring step requires human-in-the-loop evaluation and involved an expert (different from the three experts that are part of TUTORSS). Overall, each technique is evaluated for 36 unique scenarios in STUDENTSYN—we selected 18 scenarios per reference task by first picking one of the 3 target tasks and then picking a student from one of the 6 different types of behavior. The final performance results in Table 1 are re-

Method	Generative Performance		Discriminative Performance		Required Inputs and Domain Knowledge				
	Reference task T^4	Reference task T^{18}	Reference task T^4	Reference task T^{18}	Ref. task dataset: student attempts	Ref. task dataset: similar tasks	Student types	Expert grammars	Expert evaluation
RANDD	1.00	1.00	10.0	10.0	-	-	-	-	-
EDITD	1.00	1.00	31.5	48.9	-	-	-	-	-
EDITEMBD	1.00	1.00	39.6	48.9	\times	-	-	-	-
NEURSS	3.00	2.83	43.8	57.2	\times	\times	-	-	-
SYMSS	3.78	3.72	88.1	62.1	-	-	\times	\times	-
TUTORSS	3.85	3.90	89.8	85.2	-	-	-	-	\times

Table 1: This table shows results on STUDENTSYN in terms of the generative and discriminative performance measures. The columns under “Required Inputs and Domain Knowledge” highlight information used by different techniques (\times indicates the usage of the corresponding input/knowledge). The values are in the range [1.0, 4.0] for generative performance and in the range [0.0, 100.0] for discriminative performance—higher values being better. Human experts (TUTORSS) can achieve high performance on both the measures, whereas simple baselines perform poorly. NEURSS and SYMSS significantly improve upon the simple baselines; yet, there is a high gap in performance in comparison to that of human experts.

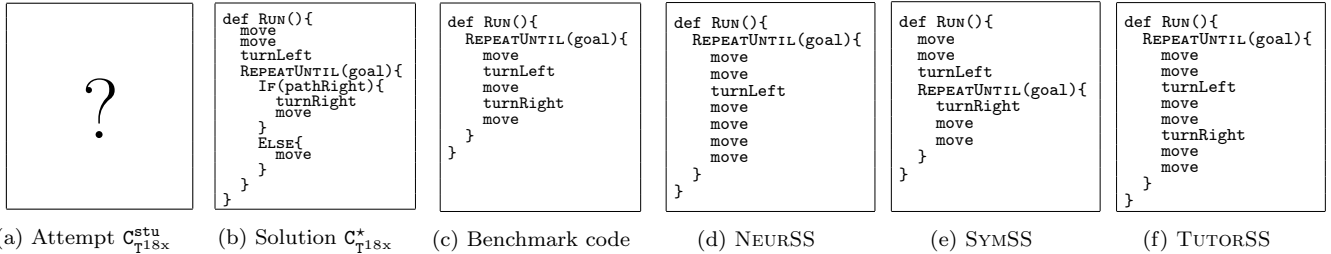


Figure 4: Qualitative results for the scenario in Figure 1. (a) The goal is to synthesize the student stu ’s behavior on the target task T^{18x} . (b) Solution code $C_{T^{18x}}^*$ for the target task. (c) Code provided in the benchmark as a possible answer for this scenario. (d, e) Codes synthesized by our techniques NEURSS and SYMSS. (f) Code provided by one of the human experts.

ported as an average across these scenarios; for TUTORSS, each of the three experts independently responded to these 36 scenarios and the final performance is averaged across experts. The simple baselines (RANDD, EDITD, EDITEMBD) have a score of 1.00 as they do not have a synthesis capability. TUTORSS achieves the highest performance; SYMSS also achieves high performance (only slightly lower than that of TUTORSS)—the high performance of SYMSS is expected given its knowledge about types of students in STUDENTSYN and the expert domain knowledge inherent in its design. NEURSS improves upon simple baselines, but performs worse compared to SYMSS and TUTORSS. Figure 4 illustrates the codes generated by different techniques for the scenario in Figure 1—the codes by TUTORSS and SYMSS are high-scoring w.r.t. our 4-point *Likert scale*; however, the code by NEURSS only captures elements of the student’s behavior in $C_{T^{18x}}^{stu}$ but misses elements of the target task T^{tar} . We provide additional details and statistical significance results w.r.t. χ^2 test [6] in the longer version of the paper [47].

Discriminative performance. As discussed in Section 3.2, we evaluate the discriminative performance of a technique in the following steps: (a) a discriminative instance is created with a scenario (T^{ref} , $C_{T^{ref}}^{stu}$, T^{tar} , ?) picked from the benchmark and 10 code options created automatically; (b) the technique chooses one of the options as stu ’s attempt; (c) the chosen option is scored either 100.0 when correct, or 0.0 otherwise. For all techniques except TUTORSS, we perform evaluation on a set of 720 instances (360 instances per reference task); for TUTORSS, we perform evaluation on a small set of 72 instances (36 instances per reference task), to reduce the effort for human experts. The final performance results in Table 1 are reported as an average predic-

tive accuracy across the evaluated instances; for TUTORSS, each of the three experts independently responded to the instances and the final performance is averaged across experts. Results highlight the huge performance gap between the human experts (TUTORSS) and simple baselines (RANDD, EDITD, EDITEMBD). Our proposed techniques (NEURSS and SYMSS) have substantially reduced this performance gap w.r.t. TUTORSS. SYMSS achieves high performance compared to simple baselines and NEURSS; moreover, on the reference task T^4 , its performance is close to that of TUTORSS. The high performance of SYMSS is partly due to its access to types of students in STUDENTSYN; in fact, this information is used only by SYMSS and is not even available to human experts in TUTORSS (see column “Student types” in Table 1). NEURSS outperformed simple baselines but its performance is below SYMSS and TUTORSS. We provide additional details and statistical significance results w.r.t. Tukey’s HSD test [49] in the longer version of the paper [47].

6. CONCLUSIONS

We investigated student modeling in the context of block-based visual programming environments, focusing on the ability to automatically infer students’ misconceptions and synthesize their expected behavior. We introduced a novel benchmark, STUDENTSYN, to objectively measure the generative as well as the discriminative performance of different techniques. The gap in performance between human experts (TUTORSS) and our techniques (NEURSS, SYMSS) highlights the challenges in synthesizing student attempts for programming tasks. We believe that the benchmark will facilitate further research in this crucial area of student modeling for block-based visual programming environments.

7. ACKNOWLEDGMENTS

This work was supported in part by the European Research Council (ERC) under the Horizon Europe programme (ERC StG, grant agreement No. 101039090).

References

- [1] U. Z. Ahmed, M. Christakis, A. Efremov, N. Fernandez, A. Ghosh, A. Roychoudhury, and A. Singla. Synthesizing Tasks for Block-based Programming. In *NeurIPS*, 2020.
- [2] F. Ai, Y. Chen, Y. Guo, Y. Zhao, Z. Wang, G. Fu, and G. Wang. Concept-Aware Deep Knowledge Tracing and Exercise Recommendation in an Online Learning System. In *EDM*, 2019.
- [3] R. Bunel, M. J. Hausknecht, J. Devlin, R. Singh, and P. Kohli. Leveraging Grammar and Reinforcement Learning for Neural Program Synthesis. In *ICLR*, 2018.
- [4] X. Chen, C. Liu, and D. Song. Execution-Guided Neural Program Synthesis. In *ICLR*, 2019.
- [5] N. Chomsky. On Certain Formal Properties of Grammars. *Information and control*, 2:137–167, 1959.
- [6] W. G. Cochran. The χ^2 Test of Goodness of Fit. *The Annals of Mathematical Statistics*, pages 315–345, 1952.
- [7] J. Cock, M. Marras, C. Giang, and T. Käser. Early Prediction of Conceptual Understanding in Interactive Simulations. In *EDM*, 2021.
- [8] Code.org. Code.org – Learn Computer Science. <https://code.org/>.
- [9] Code.org. Hour of Code – Classic Maze Challenge. <https://studio.code.org/s/hourofcode>.
- [10] Code.org. Hour of Code Initiative. <https://hourofcode.com/>.
- [11] A. T. Corbett and J. R. Anderson. Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. *User Modeling and User-Adapted Interaction*, 4(4):253–278, 1994.
- [12] A. T. Corbett, M. McLaughlin, and K. C. Scarpinato. Modeling Student Knowledge: Cognitive Tutors in High School and College. *User Model. User Adapt. Interact.*, 2000.
- [13] J. Devlin, R. Bunel, R. Singh, M. J. Hausknecht, and P. Kohli. Neural Program Meta-Induction. In *NeurIPS*, 2017.
- [14] J. Devlin, J. Uesato, S. Bhupatiraju, R. Singh, A. Mohamed, and P. Kohli. Robustfill: Neural Program Learning under Noisy I/O. In D. Precup and Y. W. Teh, editors, *ICML*, 2017.
- [15] A. Efremov, A. Ghosh, and A. Singla. Zero-shot Learning of Hint Policy via Reinforcement Learning and Program Synthesis. In *EDM*, 2020.
- [16] K. Ellis, M. I. Nye, Y. Pu, F. Sosa, J. Tenenbaum, and A. Solar-Lezama. Write, Execute, Assess: Program Synthesis with a REPL. In *NeurIPS*, 2019.
- [17] K. Ellis, C. Wong, M. I. Nye, M. Sablé-Meyer, L. Cary, L. Morales, L. B. Hewitt, A. Solar-Lezama, and J. B. Tenenbaum. Dreamcoder: Growing Generalizable, Interpretable Knowledge with Wake-Sleep Bayesian Program Learning. *CoRR*, abs/2006.08381, 2020.
- [18] A. Emerson, A. Smith, F. J. Rodríguez, E. N. Wiebe, B. W. Mott, K. E. Boyer, and J. C. Lester. Cluster-Based Analysis of Novice Coding Misconceptions in Block-Based Programming. In *SIGCSE*, 2020.
- [19] A. Ghosh, S. Tschischek, S. Devlin, and A. Singla. Adaptive Scaffolding in Block-based Programming via Synthesizing New Tasks as Pop Quizzes. In *AIED*, 2022.
- [20] S. Gulwani, O. Polozov, and R. Singh. Program Synthesis. *Foundations and Trends® in Programming Languages*, 2017.
- [21] J. He-Yueya and A. Singla. Quizzing Policy Using Reinforcement Learning for Inferring the Student Knowledge State. In *EDM*, 2021.
- [22] A. Hunziker, Y. Chen, O. M. Aodha, M. G. Rodriguez, A. Krause, P. Perona, Y. Yue, and A. Singla. Teaching Multiple Concepts to a Forgetful Learner. In *NeurIPS*, 2019.
- [23] T. Käser and D. L. Schwartz. Modeling and Analyzing Inquiry Strategies in Open-Ended Learning Environments. *Journal of AIED*, 30(3):504–535, 2020.
- [24] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum. Human-level Concept Learning through Probabilistic Program Induction. *Science*, 2015.
- [25] Y. Li, D. Choi, J. Chung, N. Kushman, J. Schrittwieser, R. Leblond, J. Keeling, F. Gimeno, A. D. Lago, T. Hubert, P. Choy, and C. de. Competition-Level Code Generation with AlphaCode. 2022.
- [26] A. Malik, M. Wu, V. Vasavada, J. Song, M. Coots, J. Mitchell, N. D. Goodman, and C. Piech. Generative Grading: Near Human-level Accuracy for Automated Feedback on Richly Structured Problems. In *EDM*, 2021.
- [27] J. C. Martin. *Introduction to Languages and the Theory of Computation*, volume 4. McGraw-Hill NY, 1991.
- [28] R. McIlroy-Young, S. Sen, J. M. Kleinberg, and A. Anderson. Aligning Superhuman AI with Human Behavior: Chess as a Model System. In *KDD*, 2020.
- [29] R. McIlroy-Young and R. Wang. Detecting Individual Decision-Making Style: Exploring Behavioral Stylometry in Chess. In *NeurIPS*, 2021.
- [30] M. I. Nye, A. Solar-Lezama, J. Tenenbaum, and B. M. Lake. Learning Compositional Rules via Neural Program Synthesis. In *NeurIPS*, 2020.
- [31] B. Paaßen, B. Hammer, T. W. Price, T. Barnes, S. Gross, and N. Pinkwart. The Continuous Hint Factory - Providing Hints in Continuous and Infinite Spaces. *Journal of Educational Data Mining*, 2018.

- [32] E. Parisotto, A. Mohamed, R. Singh, L. Li, D. Zhou, and P. Kohli. Neuro-Symbolic Program Synthesis. In *ICLR*, 2017.
- [33] C. Piech, J. Bassen, J. Huang, S. Ganguli, M. Sahami, L. J. Guibas, and J. Sohl-Dickstein. Deep Knowledge Tracing. In *NeurIPS*, pages 505–513, 2015.
- [34] C. Piech, J. Huang, A. Nguyen, M. Phulsuksombati, M. Sahami, and L. J. Guibas. Learning Program Embeddings to Propagate Feedback on Student Code. In *ICML*, 2015.
- [35] C. Piech, M. Sahami, J. Huang, and L. J. Guibas. Autonomously Generating Hints by Inferring Problem Solving Policies. In *L@S*, 2015.
- [36] L. Portnoff, E. N. Gustafson, K. Bicknell, and J. Rollinson. Methods for Language Learning Assessment at Scale: Duolingo Case Study. In *EDM*, 2021.
- [37] T. W. Price and T. Barnes. Position paper: Block-based Programming Should Offer Intelligent Support for Learners. In *2017 IEEE Blocks and Beyond Workshop (B B)*, 2017.
- [38] T. W. Price, Y. Dong, and D. Lipovac. iSnap: Towards Intelligent Tutoring in Novice Programming Environments. In *SIGCSE*, pages 483–488, 2017.
- [39] T. W. Price, R. Zhi, and T. Barnes. Evaluation of a Data-driven Feedback Algorithm for Open-ended Programming. *EDM*, 2017.
- [40] A. N. Rafferty, R. Jansen, and T. L. Griffiths. Using Inverse Planning for Personalized Feedback. In *EDM*, 2016.
- [41] M. Resnick, J. Maloney, A. Monroy-Hernández, N. Rusk, E. Eastmond, K. Brennan, A. Millner, E. Rosenbaum, J. Silver, B. Silverman, et al. Scratch: Programming for All. *Communications of the ACM*, 2009.
- [42] B. Settles and B. Meeder. A Trainable Spaced Repetition Model for Language Learning. In *ACL*, 2016.
- [43] A. Shakya, V. Rus, and D. Venugopal. Student Strategy Prediction using a Neuro-Symbolic Approach. *EDM*, 2021.
- [44] Y. Shi, K. Shah, W. Wang, S. Marwan, P. Penmetsa, and T. W. Price. Toward Semi-Automatic Misconception Discovery Using Code Embeddings. In *LAK*, 2021.
- [45] R. Singh, S. Gulwani, and A. Solar-Lezama. Automated Feedback Generation for Introductory Programming Assignments. In *PLDI*, pages 15–26, 2013.
- [46] A. Singla, A. N. Rafferty, G. Radanovic, and N. T. Hefernan. Reinforcement Learning for Education: Opportunities and Challenges. *CoRR*, abs/2107.08828, 2021.
- [47] A. Singla and N. Theodoropoulos. From {Solution Synthesis} to {Student Attempt Synthesis} for Block-Based Visual Programming Tasks. *CoRR*, abs/2205.01265, 2022.
- [48] D. Trivedi, J. Zhang, S. Sun, and J. J. Lim. Learning to Synthesize Programs as Interpretable and Generalizable policies. *CoRR*, abs/2108.13643, 2021.
- [49] J. W. Tukey. Comparing Individual Means in the Analysis of Variance. *Biometrics*, 5 2:99–114, 1949.
- [50] L. Wang, A. Sy, L. Liu, and C. Piech. Learning to Represent Student Knowledge on Programming Exercises Using Deep Learning. In *EDM*, 2017.
- [51] D. Weintrop and U. Wilensky. Comparing Block-based and Text-based Programming in High School Computer Science Classrooms. *ACM Transactions of Computing Education*, 18(1):1–25, 2017.
- [52] M. Wu, M. Mosse, N. D. Goodman, and C. Piech. Zero Shot Learning for Code Education: Rubric Sampling with Deep Learning Inference. In *AAAI*, 2019.
- [53] J. Yi, U. Z. Ahmed, A. Karkare, S. H. Tan, and A. Roychoudhury. A Feasibility Study of Using Automated Program Repair for Introductory Programming Assignments. In *ESEC/FSE*, 2017.

Mining Assignment Submission Time to Detect At-Risk Students with Peer Information

Yuancheng Wang¹, Nanyu Luo¹, Jianjun Zhou^{2*}

¹ The Chinese University of Hong Kong, Shenzhen, China

² Shenzhen Research Institute of Big Data, Shenzhen, China

{yuanchengwang, nanyuluo}@link.cuhk.edu.cn, zhoujianjun@cuhk.edu.cn

ABSTRACT

Doing assignments is a very important part of learning. Students' assignment submission time provides valuable information on study attitudes and habits which strongly correlate with academic performance. However, the number of assignments and their submission deadlines vary among university courses, making it hard to use assignment submission time as a feature to predict academic performance. In this paper, we propose a new method called Relative Assignment Submission Time (RAST) which uses the assignment submission information of peer students to improve the correlation with course grades. Experiments on real-life data of 20 courses show that RAST has a high correlation with students' academic performance. We also build a machine learning model using RAST as a feature to detect students who would suffer from poor grades. Our method outperforms the traditional method by up to 61% on f1-score. We believe that our proposed method can help other studies on assignment submission time to improve the prediction accuracy on academic performance and detecting at-risk students.

Keywords

assignment submission time, peer information, academic performance, machine learning, students at-risk

1. INTRODUCTION

Doing assignments is a traditional method of testing the quality of students' learning and consolidating what they have learned. Typically, instructors assign some exercises or projects to students at the end of a class, and then in most cases students are required to submit their assignments by a certain deadline. Many studies have shown that there is a strong correlation between assignment completion and academic performance in the corresponding courses [7, 20, 21]. Some studies also show that the time when students start doing assignments can also reflect students' learning

psychology [1]. Generally speaking, the time when students start doing assignments is related to students' procrastination psychology. If a student suffers from serious procrastination, the student is likely to choose to start doing assignments close to the deadlines, which could result in late or poor-quality submissions. A meta-analysis [11] on 33 studies confirms that procrastination is negatively correlated with academic performance.

Many studies have demonstrated the impact of students' assignment behavior (including time spent on assignments, help-seeking behavior, etc.) on academic performance in online learning environments [7, 20, 21]. In most offline learning environments, however, it is hard to know exactly when students start working on their assignments, how they work out the solution, and how much time they spend on the assignments, etc. The difference between online and offline settings makes the methodologies in many studies limited to online teaching platforms only. In most offline learning environments, the only data on assignment behavior is the submission time.

Different from other continuous learning behavior data such as class attendance, assignment submission time could have distinctive implications before and after the corresponding deadline. In most learning environments, late submissions of assignments receive penalties on grading, which makes the deadline a watershed on student behavior. The straightforward way of using the difference between submission time and its deadline cannot capture all the information of submission time. In this paper, we propose an approach to process late and normal submissions separately while still using submission time data as a single continuous fashion, so that the new variable can be easily integrated into machine learning prediction methods that require continuous features.

The popularity of Learning Management Systems (LMS) makes it possible to collect data and study multiple courses at the same time; however, it is still challenging to apply analysis on submission time data from multiple courses. Most studies in the literature focus on using the data from one course only. The key issue is that cross-course data analysis is fundamentally different across different courses. The numbers of assignments in different courses are different. More assignments usually mean less time to work out the solution, so that it is hard to analyze the submission time of different courses together. Furthermore, the difficulty levels

*The corresponding author.

Y. Wang, N. Luo, and J. Zhou. Mining assignment submission time to detect at-risk students with peer information. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 462–469, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853175>

of assignments may vary a lot. Even with the same number of assignments, courses with difficult assignments may have larger numbers of late submissions. Students of course A with difficult assignments may tend to submit assignments closer to the deadlines than other students, but this does not necessarily mean that students of A are more procrastinating. Inspired by a recent study [5] that compares attendance rates using peer attendance data, this study proposes a new indicator on assignment submission time that incorporates peer information so that submission time data from multiple courses can be compared and used in a machine learning model. So far as we know, there is no previous study that applies peer information on assignment submission time.

In this paper, we propose a new indicator called Relative Assignment Submission Time (RAST), which reflects the average distance between a student's assignment submission time and the corresponding deadline, while using peer assignment submission information to eliminate differences in time limits and difficulty levels of assignments. Our approach treats late and normal submissions in different groups but computes the new indicator as a single continuous variable, so that it can be easily applied to most machine learning methods as a prediction feature. We investigate the correlation between RAST and academic performance on real-life data from 5679 samples of 20 courses, showing that there is a high Pearson correlation between RAST and course grades. We also use RAST and other commonly available features to build models to detect at-risk students. Using the data from the first half of the semester, our method outperforms the traditional method by up to 61% on f1-score. We believe that our method can help other studies on assignment submission time to improve the prediction accuracy on academic performance and detecting at-risk students.

The rest of the paper is organized as follows. Section 2 discusses related work. Section 3 describes the preliminary. Section 4 introduces the method we propose, including the definition of Relative Assignment Submission Time (RAST). Section 5 presents the experimental results on real-life data and the analysis of the experimental results. Section 6 discusses the limitations of this paper and the future work. Section 7 concludes the paper.

2. RELATED WORK

Assignments are very important to learning. With the development of online education over the last decade, many platforms and methods have been proposed to help people collect and analyze students' assignment information. Jivet et al. [10] designed a learning analytics dashboard for Coursera MOOCs which can record when learners submit assignments. Studies in [2, 25] used clickstream data (including submitting assignments) in online teaching platforms to analyze students' behavior. [3] predicted the decrease of students' engagement in typical MOOC tasks such as watching lecture videos or submitting assignments.

Many studies have shown a correlation between students' assignment completion and procrastination. Studies in [19] analyzed student self-discipline by tracking student behavior including assignment submission. [17] comprehensively studied the relationship between students' learning behaviors and procrastination. [1] proposed a new measure called

Procrastination Index, which represents a learner's degree of procrastination in the start time of doing assignments. [24] studied the relationship between procrastination and assignment deadlines. The authors showed that dependencies between students' historical activities and the future ones provide meaningful interpretations in terms of students' procrastination behaviors.

Many studies have analyzed the relationship between assignment and academic performance. [7] explored the effects of help-seeking behavior on learning assignments to academic performance. [20] analyzed the relation between consistency in students' online work habits (including assignment submissions) and academic performance in a blended course. [21] confirmed the positive correlation between the time students spend on assignments and their academic performance.

An important topic in educational data mining is to identify and help at-risk students. Studies in [6] proposed a classifier to identify students at-risk for disengagement from coursework. [8] proposed a solution in the absence of data from previous courses to identify students at risk of failing the course. [13] developed a machine learning framework to conduct accurate at-risk student identification specialized in K-12 multimodal online environments. [15] identified at-risk students based on their behavior of handing in assignments on online platforms. Using features of attendance rate, grade retention and student profile, [23, 14] established early warning systems to identify at-risk students.

Recently, [4, 16, 12] studied the impact of assignment submission time on academic performance. [4] used students' precise assignment submission time to identify at-risk students, but without introducing peer information from the classmates. The authors suggested that assignment submission time is not easy to predict at-risk students, as the relationship between submission time and marks varies depending on both the student and the assignment context. [16] studied patterns in assignment submission time. By building a machine learning model, the authors found that completion time performs better in predicting students' assignment grades than quizzes and exam grades in a one-semester physics course with 1374 students. [12] applied the assignment submission time with temporal learning analytics to aid precision education. The authors analyzed the transitional patterns between successive assignments with Markov chains and the relationship between the patterns and the passes with association rules from 69 students.

[5] proposed a method called Relative Attendance Index to measure attendance rates, which reflects students' efforts on attending courses. While traditional attendance focuses on the record of a single person or course, relative attendance emphasizes peer attendance information of relevant individuals or courses, making the comparisons of attendance more justified. However, due to the different nature of the data (attendance vs. submission time) the specific method of introducing peer information in that paper cannot be applied to assignment submission time.

None of the above studies applied peer information to study assignment submission time.

3. PRELIMINARY

Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable [22]. Given dataset $D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_N, y_N), \mathbf{x}_i \in \mathbb{R}^n, y_i \in \{0, 1\}, i = 1, 2, \dots, N\}$, the loss function to minimize is the cross entropy loss function below.

$$L(\mathbf{w}, b) = \frac{1}{N} \left(\sum_{i=1}^N y_i \ln f(\mathbf{x}_i) + (1 - y_i) \ln(1 - f(\mathbf{x}_i)) \right). \quad (1)$$

Sometimes, we will encounter the problem that the difference between the number of positive ($y = 1$) and the number of negative ($y = 0$) samples is too large. A simple way is to use the loss function in the “**balanced**” mode [18]. The “balanced” mode uses the values of y to adjust weights in the loss function. In the “balanced” mode, the weight of $y = 0$ will be given by $N/(2 \times \text{count}(y = 0))$ and the weight of $y = 1$ will be given by $N/(2 \times \text{count}(y = 1))$. In this paper, we will refer to the original method without adjusting the loss function as the “**unbalanced**” mode.

4. METHOD

To give the definition of Relative Assignment Submission Time (RAST), we need to define some quantities first.

How long a student submits an assignment before the deadline is an indicator of study habits and attitude, we define it as Submission Buffer.

Definition 1 (Submission Buffer $\Delta_{c,s,i}$): Given course c and student s , let $\mu_{c,i}$ be the deadline for submission of the i th assignment of course c ; let $t_{c,s,i}$ be the submission time of the i th assignment of course c for student s . Then the Submission Buffer is defined as below.

$$\Delta_{c,s,i} = \mu_{c,i} - t_{c,s,i}. \quad (2)$$

The average assignment submission time of a student in a course can be defined straight-forwardly.

Definition 2 (Average Submission Time $AST_{c,s}$): Given course c and student s , let n_c be the number of assignments in c , then the Average Submission Time (AST) of c and s is defined as below.

$$AST_{c,s} = \frac{1}{n_c} \sum_{i=1}^{n_c} \Delta_{c,s,i}. \quad (3)$$

One drawback of AST is that it mixes the data of late submissions with normal submissions together. Late submission reflects the punctuality and study habits of the students. If a student has large submission buffers in some assignments, late submission information (even if it is a frequent pattern) will be buried in the average of submission buffers. Therefore, in our method we separate normal submissions and late submissions in the calculation of the new index.

Positive and negative submission buffers indicate normal and late submissions respectively.

Definition 3 (Normal Set $S_{c,i}^+$ and Late Set $S_{c,i}^-$): Given the definition of submission buffer $\Delta_{c,s,i}$, let S_c be the set of

students who registered for course c , then normal set ($S_{c,i}^+$) and late set ($S_{c,i}^-$) represent normal and late submissions.

$$S_{c,i}^+ = \{s : \Delta_{c,s,i} \geq 0 | s \in S_c\},$$

$$S_{c,i}^- = \{s : \Delta_{c,s,i} \leq 0 | s \in S_c\}.$$

With the definitions of Normal Set and Late Set, we further define a notion called Submission Deviant to reflect how a submission is different from submissions of other students in the same course in terms of submission time. We process normal and late submissions in their own category to preserve the difference between late and on-time submissions. For an assignment submission, Submission Deviant is defined as the submission buffer divided by the largest submission buffer of the same category (late or on-time). The peer information of other students’ submission time helps normalizing the significance of a late or normal submission. For example, if a student submits an assignment early but not that early when compared with the best student in the class in terms of submission buffer (who has the earliest submission), the significance of this early submission is low; similarly, if a student submits an assignment late but not that late when compared with the worst student in the class, the significance of this late submission is also low. Formally, Submission Deviant is defined as below.

Definition 4 (Submission Deviant $d_{c,s,i}$): The Submission Deviant of the i th assignment submission of student s in course c is defined by the formula below.

$$d_{c,s,i} = \begin{cases} \frac{\Delta_{c,s,i}}{\max_{s' \in S_{c,i}^+} |\Delta_{c,s',i}|} & \Delta_{c,s,i} \geq 0 \\ \frac{\Delta_{c,s,i}}{\max_{s' \in S_{c,i}^-} |\Delta_{c,s',i}|} & \Delta_{c,s,i} \leq 0 \end{cases} \quad (4)$$

By the definition, normal and late submissions have positive and negative submission deviants respectively.

Finally, Relative Assignment Submission Time (RAST) is defined as the average submission deviant over all assignments of a course for each student.

Definition 5 (Relative Assignment Submission Time $RAST_{c,s}$): Given course c and student s , the Relative Assignment Submission Time (RAST) is defined as below.

$$RAST_{c,s} = \frac{1}{n_c} \sum_{i=1}^{n_c} d_{c,s,i}. \quad (5)$$

Relative Assignment Submission Time (RAST) reflects the average distance between a student’s assignment submission time and the corresponding deadline, while using peer assignment submission information to eliminate differences in time limits and difficulty levels of assignments. Compared with Relative Assignment Submission Time (RAST), Assignment Submission Time (AST) in Definition 2 does not consider peer assignment submission information.

Lemma 1: Given student s in course c and the Relative Assignment Submission Time $RAST_{c,s}$, $RAST_{c,s} \in [-1, 1]$.

Table 1: Courses in The Experiment

Course code	Course title
CSC1001	Programming Methodology
CSC1002	Computational Laboratory
CSC3001	Discrete Mathematics
CSC3002	Programming Paradigms
CSC4020	Fundamentals of Machine Learning
EIE2050	Digital Logic and Systems
ERG2050	Introduction to Data Analytics
FIN2020	Foundation of Finance
MAT2002	Ordinary Differential Equations
MAT2007	Elementary Real Analysis II
MAT2040	Linear Algebra
MAT3007	Optimization
MAT3040	Advanced Linear Algebra
MAT3253	Complex Variables
STA2001	Probability and Statistics I
STA3010	Regression Analysis
STA3020	Statistical Inference
STA3100	Advanced Statistics

PROOF. The proof is straightforward. \square

Students who are always submitting assignments before classmates (and ahead of the deadlines) will have an RAST of 1, while students who are always missing the deadlines and are the last one to submit assignments have an RAST of -1. The fixed range of $[-1, 1]$ provides the minimal and maximal values of RAST, which can make the data analysis more convenient.

5. RESULTS

5.1 Datasets

The anonymous data used in this paper were collected from a university in China in the Fall 2020 semester and the Spring 2021 semester. Note that all data collection and usage in this project have been approved by the university management. To protect students’ privacy, all student IDs were encrypted and processed into hash codes, with other data related to a student being linked by the corresponding hash code. The data came from 20 undergraduate courses (15 in the Spring 2021 semester and 5 courses in the Fall 2020 semester). As shown in Table 1, these courses covered multiple topics on Computer Science (CSC), Math (MAT), and Statistics (STA). MAT2040 and MAT3007 were offered in both semesters. We collected the deadline for each assignment in a course and then the submission time of each assignment for each student from the LMS system “Blackboard”. In all the courses, students received letter grades from A to F at the end of the courses. The corresponding relationship between letter grades and numerical grades is shown in Table 2. The proportions of different grades in different courses were not exactly the same, but the differences were not very significant due to the grading quality control of the university. Many courses had the policy of not accepting submissions of 4 or more days late. For the simplicity of data reprocessing, we considered missing assignments as 3-day late. These samples only account for 4.2% of all the 5679 samples.

Table 2: Corresponding Relationship between Grades and Numeric Grades

Letter grade	A	A-	B+	B	B-	C+
Numerical grade	4.0	3.7	3.3	3.0	2.7	2.3
Letter grade	C	C-	D+	D	D-	F
Numerical grade	2.0	1.7	1.3	1.0	0.7	0

Table 3: Correlation Coefficient between RAST/AST and Numeric Grades

course	enrollment	AST	RAST
MAT3253 (21 spring)	76	0.69	0.73
MAT3007 (21 spring)	150	0.71	0.73
MAT2002 (21 spring)	212	0.56	0.72
CSC1001 (21 spring)	1037	0.39	0.69
CSC3001 (20 fall)	301	0.47	0.67
MAT2040 (20 fall)	452	0.51	0.65
MAT2040 (21 spring)	184	0.59	0.63
STA2001 (21 spring)	818	0.53	0.62
MAT2007 (21 spring)	107	0.54	0.61
CSC4020 (21 spring)	103	0.45	0.60
MAT3007 (20 fall)	343	0.41	0.58
CSC1002 (21 spring)	930	0.38	0.57
EIE2050 (20 fall)	121	0.45	0.57
CSC3002 (20 fall)	151	0.32	0.54
ERG2050 (21 spring)	82	0.46	0.52
STA3010 (21 spring)	217	0.48	0.46
STA3100 (21 spring)	100	0.27	0.35
STA3020 (21 spring)	100	0.36	0.32
FIN2020 (21 spring)	157	0.25	0.29
MAT3040 (21 spring)	56	0.21	0.25

5.2 Correlation with Academic Performance

Many previous studies showed that assignment submission is correlated with academic performance. We calculated the Pearson correlation between students’ Relative Assignment Submission Time (RAST) and their course numeric grades across 20 courses. As a comparison, we also calculated the Pearson correlation between the corresponding Average Submission Time (AST) and the course numeric grades. The results are shown in Table 3 (sorted by the correlation coefficient of RAST). The second column of the table is the course enrollment number, the third column is the AST correlation coefficient, and the fourth column is the RAST correlation coefficient. The correlation coefficients between RAST and course grades are higher than that of AST across 18 courses. The p-values of all the correlation coefficients in Table 3 are all well below 0.005, which shows that the correlation coefficients are statistically significant [9].

5.3 RAST Distribution

In order to show the different distributions on RAST of high and low course grade students, we selected two sample sets from all the data in the Fall 2020 semester of our dataset, one with grades greater than or equal to A- (sample size

438), and the other with grades less than or equal to C+ (the sample size 208). Figure 1 shows the distribution on RAST of samples with grades $\geq A-$ (in blue) and the distribution on RAST of samples with grades $\leq C+$ (in grey) in Fall 2020. The data of the two sets came from five courses (CSC3001, CSC3002, MAT2040, MAT3007, and EIE2050) in Fall 2020. For samples with higher grades (in blue), the RAST values of 79% samples are greater than 0, and 13% samples are greater than 0.25. For samples with lower grades (in grey), the RAST of the 75% samples are less than 0, 39% samples are less than -0.25, and 19% samples are even less than -0.5. We can see that students with high grades have a higher probability of having high RAST, whereas students with lower grades have a higher probability of having low RAST. We did a similar experiment on Spring 2021 data (The two sample sets are of size 1422 and 705 respectively), and the results are shown in Figure 2. The distributions are similar to those in Figure 1.

5.4 Predicting Academic Performance

In this experiment, we used students’ RAST and some other attributes as features in logistic regression models to predict their academic performance. More precisely we predicted whether a student will have a struggling performance (course grade $\leq C+$). The data used in our experiments came from 20 courses. For each sample $\{x, y\}$, x is a vector consisting of 4 features, including the RAST in the course, average GPA in previous semesters, year in university, and the school (in one-hot coding). $y = 1$ if grade $\leq C+$, or otherwise $y = 0$. Note that we used a mix of data from different courses because the grading criteria for each course are not very different and RAST is in the range of -1 to 1. We randomly divided the data into a training set and a test set according to the ratio of 4:1. We used the cross entropy loss function for Logistic Regression training in order to avoid overfitting. The total number of samples is 5697, with 15.6% of them being positive samples ($y = 1$). Since the data is imbalanced, we experimented both with and without the “balanced” mode. As discussed in the Preliminary (Section 3), the “balanced” mode can improve the recall while sacrificing the precision. All experimental results are averaged after ten repetitions. At the same time, we also tried replacing RAST with AST and conducted experiments for comparison. The results are shown in Table 4. In the unbalanced mode, RAST is better than AST on precision (0.80 vs. 0.68), while AST is better than RAST on recall (0.49 vs. 0.44). The two methods are tied on f1-score. In the “balanced” mode, however, RAST is better than AST on all the three measures (precision, recall and f1-score).

Furthermore, in order to evaluate the feature importance of RAST, we also repeated the above experiment without using RAST as a feature. The results (with standard deviation values) are shown in the Table 5. Compared with the results using RAST, the f1-score decreased by 28% (0.57 vs. 0.41) with the “unbalanced” mode, and the f1-score decreased by 0.25% (0.68 vs. 0.51) with “balanced” mode.

5.5 Application of Half Semester Data

In real-life applications, it is important to detect at-risk students long before the semester is over. Therefore, in this experiment we used only the data from the first half of the semester to perform the training. We repeated the same ex-

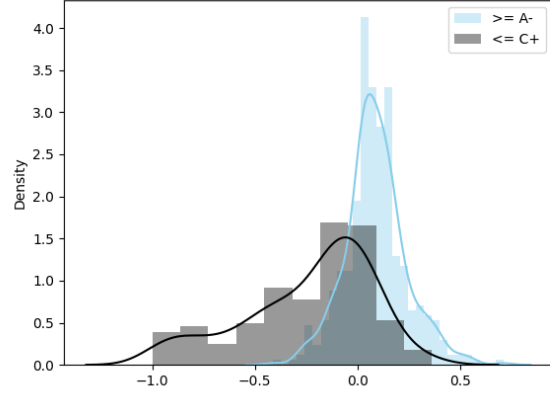


Figure 1: RAST Distributions of Fall 2020 Data

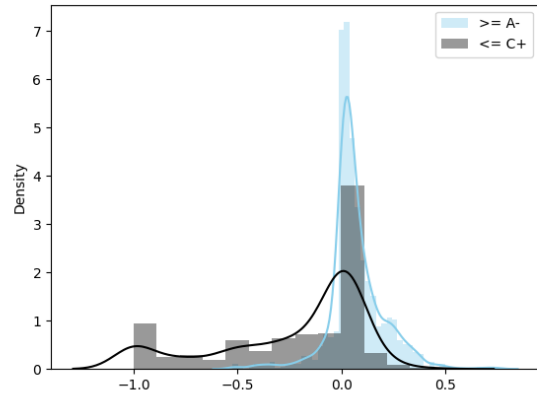


Figure 2: RAST Distributions of Spring 2021 Data

Table 4: Results of the Logistic Regression Models (with standard deviation values in the parenthesis)

RAST	“unbalanced” mode	“balanced” mode
precision	0.80 (± 0.05)	0.56 (± 0.04)
recall	0.44 (± 0.02)	0.85 (± 0.03)
f1-score	0.57 (± 0.01)	0.68 (± 0.01)
AST	“unbalanced” mode	“balanced” mode
precision	0.68 (± 0.06)	0.53 (± 0.03)
recall	0.49 (± 0.02)	0.76 (± 0.04)
f1-score	0.57 (± 0.03)	0.62 (± 0.04)

Table 5: Results of the Logistic Regression Models without Using Submission Time Features

	“unbalanced” mode	“balanced” mode
precision	0.63 (± 0.04)	0.38 (± 0.02)
recall	0.31 (± 0.05)	0.76 (± 0.04)
f1-score	0.41 (± 0.03)	0.51 (± 0.04)

Table 6: Results of the Logistic Regression Models with Half Semester Data

RAST	“unbalanced” mode	“balanced” mode
precision	0.76 (\pm 0.07)	0.52 (\pm 0.02)
recall	0.41 (\pm 0.03)	0.83 (\pm 0.05)
f1-score	0.53 (\pm 0.04)	0.64 (\pm 0.02)
AST	“unbalanced” mode	“balanced” mode
precision	0.62 (\pm 0.02)	0.52 (\pm 0.01)
recall	0.45 (\pm 0.03)	0.71 (\pm 0.02)
f1-score	0.33 (\pm 0.02)	0.56 (\pm 0.02)

periments as in the previous section, except that only the assignment data from the first half of the semester are used to train the prediction models. The results (with standard deviation values) are shown in Table 6. We can see that only using half semester data, models with RAST can still achieve better results than models with AST. In unbalanced mode, the f1-score of RAST is 61% better than that of AST (0.53 vs. 0.33). In the balanced mode, the f1-score of RAST is 14% better than that of AST (0.64 vs. 0.56). We notice that for RAST the f1-scores are only slightly worse than the original experimental results in Table 4 (0.53 vs. 0.57 for “unbalance” mode and 0.64 vs. 0.68 for “balanced” mode), even though using half amount of the data. Our experimental results inspire us to use our method to predict students who are likely to be struggling in the course halfway through the course, which allows us to take steps early to provide assistance to these students in need.

Moreover, we conducted additional experiments on individual courses, with all experimental settings unchanged. Experimental results show that our method can achieve stronger results in some individual courses, including EIE2050 (0.86 f1-score), CSC3001 (0.81 f1-score), and CSC3002 (0.80 f1-score) of Fall 2020, as well as MAT3040 (1.00 f1-score), MAT3007 (0.81 f1-score) and STA3020 (0.79 f1-score) of Spring 2021. The f1-score of other testing courses are below 0.79. See the appendix for the complete results.

6. DISCUSSION

Studies have shown that students tend to delay assessment submissions until deadlines approach [1, 3]. Our data also indicate that more than 60% of students did not submit assignments until the last 24 hours before the deadlines. The submission distribution may vary in different courses due to the assignments’ difficulties and the students’ average level. Hence, AST reflects limited information about the objectives of students. RAST takes other students in the same course into consideration. The performance of RAST better than the AST manifests that RAST is a more general feature to estimate whether a student’s assignment submission time is in a normal stage.

We have considered the circumstances that students submit their assignments after the deadlines. Even though late submissions receive heavy penalties on grading, students may be overwhelmed with the workload or forget to submit assignments, causing slightly late submissions. However, the assignments usually account for a limited proportion of the

final grade. By finishing the assignments, students practice a review and raise the chance of getting a good mark in the final exam. In this sense, late submissions are better than making no submission, and late submissions contain valuable information. Compared with the general regularization, RAST divides submissions of the same course into two groups (late and normal) to capture the information from late submissions.

In Table 3, three STA courses attain low correlation values between RAST/AST and course grades. We conjecture that course difficulty and assessment composition may influence the correlation. For example, STA3010, STA3100 and STA3020 are all advanced statistics courses. The exams are difficult, so that their final exam scores are dominant in the course grades. For such courses, assignment submission time may not be sufficient to trace students’ learning status because besides the assignments students still need to spend a lot of time reviewing and understanding the course contents. In contrast, STA2001 the elementary statistics course (this university set up statistics course codes from 2000 instead of 1000), gains a much higher correlation coefficient between RAST and course grades.

7. LIMITATIONS AND FUTURE WORK

In this paper, we studied assignments finished offline and submitted online through LMS. Since in such a setting we cannot collect the time the students start doing the assignments, the submission time may not always reflect students’ study attitudes. For example, a student may complete an assignment early, but submit it close to the deadline.

Although we mentioned in the paper that the grading standards of the courses included in the data used in this paper are not very different, in other scenarios, if the evaluation standards of different courses are very different, it may be necessary to consider the difference on grading schemes between courses when using Relative Assignment Submission Time (RAST) on multiple course data. We leave the study of such scenarios to future work.

8. CONCLUSION

Students’ assignment submission time is an important factor on study psychology and behavior analysis. In this paper, we proposed a new indicator using peer assignment submission information called Relative Assignment Submission Time (RAST). Our experiments on real-life data showed a high correlation coefficient between RAST and students’ academic performance. Using relative assignment submission time as a feature, we built a machine learning model to predict students who are likely to suffer from poor performance in courses. The experimental results show that the prediction ability of the model is significantly improved by adding RAST as a feature. We also used half semester data to train prediction models which allow us to identify students who may be struggling in the course earlier and make it easier for schools and teachers to take relevant measures in time to help these students.

9. ACKNOWLEDGMENTS

This work was supported by Shenzhen Research Institute of Big Data. We also want to thank anonymous reviewers for helpful suggestions.

10. REFERENCES

- [1] L. Agnihotri, R. Baker, and S. Stalzer. A procrastination index for online learning based on assignment start time. In *Proceedings of The 13th International Conference on Educational Data Mining (EDM20)*, pages 550–554. International Educational Data Mining Society, 2020.
- [2] N.-J. Akpınar, A. Ramdas, and U. Acar. Analyzing student strategies in blended courses using clickstream data. In *Proceedings of The 13th International Conference on Educational Data Mining (EDM20)*, pages 6–17. International Educational Data Mining Society, 2020.
- [3] M. L. Bote-Lorenzo and E. Gómez-Sánchez. Predicting the decrease of engagement indicators in a mooc. In *Proceedings of the 7th international learning analytics and knowledge conference*, pages 143–147, 2017.
- [4] S. H. Cormack, L. A. Eagle, and M. S. Davies. A large-scale test of the relationship between procrastination and performance using learning analytics. *Assessment & Evaluation in Higher Education*, 45(7):1046–1059, 2020.
- [5] P. Deng, J. Zhou, J. Lyu, and Z. Zhao. Assessing attendance by peer informations. In *Proceedings of The 14th International Conference on Educational Data Mining (EDM21)*, pages 400–406. International Educational Data Mining Society, 2021.
- [6] J. Feild, N. Lewkow, S. Burns, and K. Gebhardt. A generalized classifier to identify online learning tool disengagement at scale. In *Proceedings of the 8th International Conference on Learning Analytics and Knowledge*, pages 61–70, 2018.
- [7] A. Gurung, A. F. Botelho, and N. T. Heffernan. Examining student effort on help through response time decomposition. In *Proceedings of the 11th International Learning Analytics and Knowledge Conference*, pages 292–301, 2021.
- [8] M. Hlosta, Z. Zdrahal, and J. Zendulka. Ouroboros: early identification of at-risk students without models based on legacy data. In *Proceedings of the 7th international learning analytics and knowledge conference*, pages 6–15, 2017.
- [9] J. P. Ioannidis. The proposal to lower p value thresholds to .005. *Jama*, 319(14):1429–1430, 2018.
- [10] I. Jivet, J. Wong, M. Scheffel, M. Valle Torre, M. Specht, and H. Drachslér. Quantum of choice: How learners’ feedback monitoring decisions, goals and self-regulated learning skills are related. In *Proceedings of the 11th international learning analytics and knowledge conference*, pages 416–427, 2021.
- [11] K. R. Kim and E. H. Seo. The relationship between procrastination and academic performance: A meta-analysis. *Personality and Individual Differences*, 82:26–33, 2015.
- [12] M. Kokoç, G. Akçapınar, and M. N. Hasnine. Unfolding students’ online assignment submission behavioral patterns using temporal learning analytics. *Educational Technology & Society*, 24(1):223–235, 2021.
- [13] H. Li, W. Ding, and Z. Liu. Identifying at-risk k-12 students in multimodal online environments: a machine learning approach. In *Proceedings of The 13th International Conference on Educational Data Mining (EDM20)*, pages 137–147. International Educational Data Mining Society, 2020.
- [14] H. Li, C. F. Lynch, and T. Barnes. Early prediction of course grades: models and feature selection. In *Proceedings of The 11th International Conference on Educational Data Mining (EDM18)*, pages 492–495. International Educational Data Mining Society, 2018.
- [15] J. McBroom, B. Jeffries, I. Koprinska, and K. Yacef. Mining behaviours of students in autograding submission system logs. In *Proceedings of The 9th International Conference on Educational Data Mining (EDM16)*, pages 159–166. International Educational Data Mining Society, 2016.
- [16] M. Nieberding and A. F. Heckler. Patterns in assignment submission times: Procrastination, gender, grades, and grade components. *Physical Review Physics Education Research*, 17(1):013106, 2021.
- [17] J. Park, R. Yu, F. Rodriguez, R. Baker, P. Smyth, and M. Warschauer. Understanding student procrastination via mixture models. In *Proceedings of The 11th International Conference on Educational Data Mining (EDM18)*, pages 187–197. International Educational Data Mining Society, 2018.
- [18] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.
- [19] F. Salehian Kia, M. Hatala, R. S. Baker, and S. D. Teasley. Measuring students’ self-regulatory phases in lms with behavior and real-time self report. In *Proceedings of the 11th International Learning Analytics and Knowledge Conference*, pages 259–268, 2021.
- [20] V. Sher, M. Hatala, and D. Gašević. Analyzing the consistency in within-activity learning patterns in blended learning. In *Proceedings of the 10th International Learning Analytics and Knowledge Conference*, pages 1–10, 2020.
- [21] D. Tzimas and S. Demetriadis. The impact of learning analytics on student performance and satisfaction in a higher education course. In *Proceedings of The 14th International Conference on Educational Data Mining (EDM21)*, pages 654–660. International Educational Data Mining Society, 2021.
- [22] R. E. Wright. Logistic regression. 1995.
- [23] Y. Xu and K. Wilson. Early alert systems during a pandemic: A simulation study on the impact of concept drift. In *Proceedings of the 11th International Learning Analytics and Knowledge Conference*, pages 504–510, 2021.
- [24] M. Yao, S. Sahebi, and R. Feyzi Behnagh. Analyzing student procrastination in moocs: a multivariate hawkes approach. In *Proceedings of The 13th International Conference on Educational Data Mining (EDM20)*, pages 280–291. International Educational Data Mining Society, 2020.
- [25] T. Zhang, M. Taub, and Z. Chen. Measuring the impact of covid-19 induced campus closure on student self-regulated learning in physics online learning modules. In *Proceedings of the 11th International*

APPENDIX

Table 7: F1-Score for All Courses in Section 5.5

course	f1-score
MAT3040 (21 spring)	1.00
EIE2050 (20 fall)	0.86
MAT3007 (21 spring)	0.81
CSC3002 (20 fall)	0.80
CSC3001 (20 fall)	0.79
STA3020 (21 spring)	0.79
FIN2020 (21 spring)	0.75
MAT2002 (21 spring)	0.67
CSC4020 (21 spring)	0.67
CSC1001 (21 spring)	0.65
MAT2040 (20 fall)	0.65
MAT3253 (21 spring)	0.65
STA2001 (21 spring)	0.64
CSC1002 (21 spring)	0.63
STA3010 (21 spring)	0.63
STA3001 (21 spring)	0.61
MAT2007 (21 spring)	0.59
MAT3007 (20 fall)	0.53
MAT2040 (21 spring)	0.43
ERG2050 (21 spring)	0.31

Simulating Policy Changes In Prerequisite-Free Curricula: A Supervised Data-Driven Approach

Frederik Baucks, Laurenz Wiskott

Ruhr University Bochum, Computer Science Department, Institute for Neural Computation
{frederik.baucks, laurenz.wiskott}@ini.rub.de

ABSTRACT

Curriculum research is an important tool for understanding complex processes within a degree program. In particular, stochastic graphical models and simulations on related curriculum graphs have been used to make predictions about dropout rates, grades, and degree completion time. There exists, however, little research on changes in the curriculum and the evaluation of their impact. The available evaluation methods of curriculum changes assume pre-existing strict curriculum graphs in the form of directed acyclic graphs. These allow for a straightforward model-oriented probabilistic or graph topological investigation of curricula. But the existence of such graphs cannot generally be assumed. We present a novel generalizing approach in which a curriculum graph is constructed based on data, using measurable student flow. By applying a discrete event simulation, we investigate the impact of policy changes on the curriculum and evaluate our approach on a sample data set from a German university. Our method is able to create a comparably effective and individually verifiable simulation without requiring a curriculum graph. It can thus be extended to prerequisite-free curricula, making it feasible to evaluate changes to flexible curricula.

Keywords

Curriculum change, curriculum graph, student flow, discrete event simulation, degree completion time

1. INTRODUCTION

Curriculum Analytics is a recognized tool in Educational Data Mining to study the structure of curricula at a university [18]. One goal is to examine the structure of a curriculum and its influence on students' study progress. Often a graphical representation of the curriculum is formed, which is then called a curriculum graph, where vertices represent courses and edges a type of dependence, for example, a strict prerequisite. Due to the frequency of curricula with strict prerequisites, a commonly considered graph type is

the directed acyclic graph (DAG) (e.g., [1, 25]), where edges represent prerequisites between courses. For graphs of this type, probabilistic algorithms like Bayesian Networks can be applied to predict e.g. grades and dropout rates [22]. Furthermore, directed curriculum graphs offer a strict logic according to which students are guided through their degree program. This can be exploited for a discrete event simulation to simulate student flow, predict the degree completion time (DCT) [9, 8, 19], and investigate policy changes on the curriculum [14]. A discrete event simulation models the operation of a complex system in discrete time steps. Processes in the system are described by events, which can only take place at discrete time steps. This allows, for example, to simulate a redesign of the system [5]. In the context of degree programs, events are often modeled as courses or the corresponding exams, and time steps as semesters.

But not every degree program is based on a curriculum that can be translated into a DAG to be used in a discrete event simulation. For instance, such strict curricula hardly exist in Germany. In order to include a flexible curriculum and its influence in predictive methods, the relationships between courses must be obtained from the data. In Raji et al. [17] and Backenköhler et al. [3], grade correlations were used to form graphs of a prerequisite-free curriculum. The results are undirected cyclic graphs (UCG) that can be used for visualization and grade predictions using Markov approaches [21]. However, Markov-Networks suffer from the curse of dimensionality and thus small data set sizes, like the one we will introduce, are insufficient. Further, the resulting graphs contain no information about the actual order in which students take courses. To overcome these problems, one could try to use a discrete event simulation from the context of a DAG and generalize it to a non-strict curriculum using a data-generated curriculum graph. But a UCG does not offer a strict student transition logic as a DAG offers, therefore, it is not clear how students can be simulated through their degree program. Existing approaches that do not use a DAG, are limited to supply-demand modeling [20, 13] in which the supply capacities of courses and teachers and the demand of the students are simulated. As far as we know, all approaches using discrete event simulation to simulate student flow are lacking a data-driven evaluation of the simulation accuracy for individual students.

We model a data-driven directed cyclic curriculum graph (DCG) using time-dependent student flow, comparable to curriculum mining approaches [15, 7], and use the directed

F. Baucks and L. Wiskott. Simulating policy changes in prerequisite-free curricula: A supervised data-driven approach. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 470–476, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853177>

Table 1: Data Set Variables

Variable	Description
ID	Anonymized student ID
Course	Course name
Exam - Grade	Grade in range 0 – 100
Exam - Time	Exam’s term
Exam - Try	Attempt number
Exam - Credits	Credits for passing

edges to simulate a student’s transition from one course to the next. Thus, we get a logic comparable to that of strict curricula (DAG), but we can simulate and analyze flow in non-strict curricula (UCG). In this work in progress study, we build a generalizing approach to simulate student flow on a curriculum with no given prerequisite relations. This allows for simulating new policy changes, for example, increasing the proposed workload in a semester, or changing the frequency of exams. Further, it gives us the advantage of labeled data, and therefore makes individual error analysis possible, as our first results show. The main contributions of our paper can be summarized as follows:

1. Building a directed student flow graph using a small data set of an Applied Computer Science program.
2. Building a student flow-based logic and running an individually verifiable discrete event simulation of the DCT to investigate policy changes to the curriculum.

2. DATA

The data set consists of examination data from the Bachelor’s program Applied Computer Science at the Ruhr University Bochum. The years 2013-2020 were considered, as there has been no policy change to the curriculum during this period. Each entry in the data set consists of exam-related variables described in Table 1. The degree program has a workload of 180 credit points. Each of these credits corresponds to a time commitment of 30 hours. In order to complete the program, with a standard length of study of 6 semesters, 30 credits must be completed each semester. Of the 180 credits, 117 credits are in 19 compulsory courses that must be taken to earn the degree. We limit ourselves to these compulsory courses, as the remaining credits can be obtained from a wide range of available elective courses so that the data becomes too sparse. Besides the requirement to pass the courses, there are no other requirements to these courses, such as a fixed sequence or prerequisites, which is common in Germany. We were able to process data of a total of $N = 405$ students. These are students who are currently attending the university. Therefore, the number of students who have passed the compulsory courses for a given semester is decreasing significantly, as Table 3 shows. Here, the semesters are indicated in the row ‘Term’ and the number of students who have passed all courses from the associated term is indicated in the row ‘No. Students’. We see that only 33 students have passed all courses of the 5 semesters. In the future, the data set will be expanded to include students who completed or dropped out of the program. The grades in the degree program are given in percentages from 0 to 100. An exam and thus the associated course is considered passed when 50 percent is achieved. We will look at the grades in increments of 10. In this way, we

Table 2: Lowest Pass Rate Courses Regarding Group G^*

Course Name	Pass Rate
Mathematics II	0.909
Web-Engineering	0.818
Operating Systems	0.818
Database Systems	0.757

try to counteract the sparsity of the data and at the same time ensure a lower deanonymization risk. The grading scale thus goes from 1 to 10 and an exam is considered passed if the student receives a grade of 6 or higher. In terms of simulation, we will focus on the 33 students from Table 3 who successfully passed all compulsory courses and call this group of students G^* in the following.

2.1 Data Privacy and Ethics

An important issue in the use of personal data in Germany and the European Union is compliance with the applicable data protection law (General Data Protection Regulation (GDPR) [23]). The implementation of the GDPR on our data set was carried out by a third-party university office in close cooperation with the data protection officer of the University. The anonymization includes name encryption, omission of demographic information (e.g., origin and gender), and aggregation of courses that are too small ($N < 10$) into their upper course categories (e.g., specialization courses). All grades were randomized with a stochastic noise $z \in [-5, 5]$ in the 0-100 percentage scale. This ensured that the anonymity of individuals was adequately protected when processing the data according to the GDPR. Ethically, the implementation of the procedures in practice regarding the following methodology has not yet been clearly formulated. The fairness of the methods still needs to be ensured to prevent discrimination against individuals or groups at all costs.

2.2 Pass Rates, Workload and Retake Bonus

In the following, we present statistical insights of our data set that influenced the later process of simulation. At first, we define the Degree Completion Time (DCT) as the number of semesters taken to pass all 19 compulsory courses. The DCT will be the output value of the simulation, which will be called DCT simulation (DCT-SIM) in the following. Pass rates of a student should be of great influence for the DCT value. Nevertheless, our considered group G^* is so good that we can observe only very high pass rates, as Table 2 indicates. This table shows the 4 courses in which group G^* had the lowest pass rates on the first try of an exam. A larger data set would probably show more representative values. However, we aim to ensure that our methodology is also applicable to an extended data set without restrictions. In contrast to the pass rates of group G^* , the workload has an impact on the DCT of the observed group. Table 3 shows that in each semester different numbers of courses are proposed by the curriculum, where the courses also have unequal workloads. This is also reflected in the data. In Table 3 the average workload of the students of G^* in the first 5 semesters was calculated and compared to the workload proposed by the curriculum. For example, 3 compulsory courses worth 23 credits are recommended in the third semester and 2 courses worth 13 credits in the

Table 3: Basic Statistics Regarding Terms

Term	1	2	3	4	5
No. Courses	5	5	3	4	2
Workload	27	30	23	24	13
No. Students	185	80	55	35	33
Mean Workl. G^*	26.70	30.00	22.24	23.18	6.42

fifth semester. Unlike semesters 3,4 and 5, no courses other than compulsory courses are recommended in the first and second semesters. In semester 1, this creates a time deficit of 3 credits. Even the high-performing students from group G^* perceive this time delay with an average workload of 26.697 and have to compensate it later by doing more work or achieving a higher DCT. Furthermore, we will address the change of grades when students retry a failed exam. An exam may be attempted a total of three times before a student is disenrolled. Since our data set is small, we decided to determine the average grade change from the first to the second take of an exam. The difference is added as a bonus to the grade of the first attempt to simulate the grade of the second attempt. This works under the assumption that the students perform similarly much better in all courses if they take an exam for the second time. The resulting global average is an improvement of 2.622 out of 10 using all available student data.

3. METHODS

We want to find a simulation strategy that matches the Degree Completion Time (DCT) distribution of a given student group. In addition, it should be possible to evaluate policy changes to the curriculum using the DCT-SIM. In particular, we want to simulate group G^* . The DCT-SIM should remain as generalizable as possible for future changes in the data set. Therefore, we use the data of all available 405 students in each step of the DCT-SIM of group G^* . For example, we train prediction methods course by course, so we can use more data in the first semester than in the last semester. We have divided our methodology into three sections: Student Flow, DCT-SIM, and Policy Changes. The Student Flow indicates the transition rates of students from one semester to the next and forms the basis of our approach. The DCT-SIM is formulated using a data-driven logic based on the Student Flow, grades, and workload of all available students in the data set. We pass a group of students to our method and get the DCTs back as a distribution. The Policy Changes show the use of our DCT-SIM even for small groups, such as G^* .

3.1 Student Flow Network

Since no strict course prerequisites are given in the curriculum and thus no directed relationships between two courses, we must model these connections in a data-driven way. The flow network is a directed cyclic graph $G_f = (V_f, E_f, W_f)$, where V_f , E_f , and W_f are the sets of vertices, edges, and edge weights, respectively. Each vertex $v \in V_f$ represents a course. A directed edge $e \in E_f$ is drawn if students traverse from one course in one semester to another different course in the next semester. The percentage of students across all semesters who are in a vertex in a semester and now move along an edge to the next vertex in the next semester corresponds to the edge weight $w \in W_f$. The sum of the

outgoing percentage weights equals 1. A student is usually located on several vertices at the same time within one semester depending on the number of courses he/she is attending. Figure 1, generated using NetworkX [10], shows the complexity of the flow. Here the courses are arranged as recommended by the curriculum. From left to right in semester ascending order the courses in one column belong to one semester. For example, the first column corresponds to the first semester. The course vertex size corresponds to the number of students that managed to finish the course. The graph is highly connected (329 of 361 possible edges) because in a lot of courses students violate the recommended curriculum. The darkness and thickness of edges indicate the weight of the flow. If an edge $e_{u,v} \in E_f$ between two vertices $u, v \in V_f$ corresponds to the weight value $w_{u,v} = 1$, it means that all students who attended course u in one semester attended course v in the next semester. The edges in Figure 1 suggest that the flow for the majority of students is from left to right according to the recommended curriculum. From the later semesters, when the amount of data decreases, the flow starts to become more chaotic, for example students move from right to left. This suggests that the observed students adhere less to the study plan as they progress in the degree program.

3.2 Degree Completion Time Simulation

The DCT-SIM consists of three steps: Initialization, course selection, and pass/fail prediction. The DCT-SIM of a student is completed, when all 19 courses are passed. In one DCT-SIM run, we sample every student of group G^* exactly once.

3.2.1 Initialization

We initialize each student of G^* by having him/her attend a random subset of all the courses recommended for the first semester up to the workload that was extracted from the data of that specific student. If a student has a workload in the first semester, that exceeds the workload of all first semester courses, we assign second semester courses randomly until the workload is filled. We identified a statistically significant correlation between the grades of the courses in the first semester. We intended to include the correlation via a multivariate Gaussian distribution in the initialization. Unfortunately, the grade distributions of the courses are multi-modal. Therefore, we sample from the grade combinations of the first semester given by the data of the specific student that gets simulated. In that way, we can incorporate the grade correlations of the first semester courses as well.

3.2.2 Credit Workload and Course Selection

Various factors play a role when a student chooses the workload for an upcoming semester [12]. When simulating students from group G^* , we draw the workload from the data of the specific student that gets simulated. If a workload of 0 is drawn, it is randomly re-drawn from the normal distribution with the mean and the standard deviation of the workloads of other students from group G^* in that semester. This happens, for example, when a student has not taken any courses in a semester. If the DCT-SIM leads to semesters in which no data are available, a workload of 15 credits is set. Next, we need to select courses to fulfill the chosen workload. Courses are randomly drawn based on the student

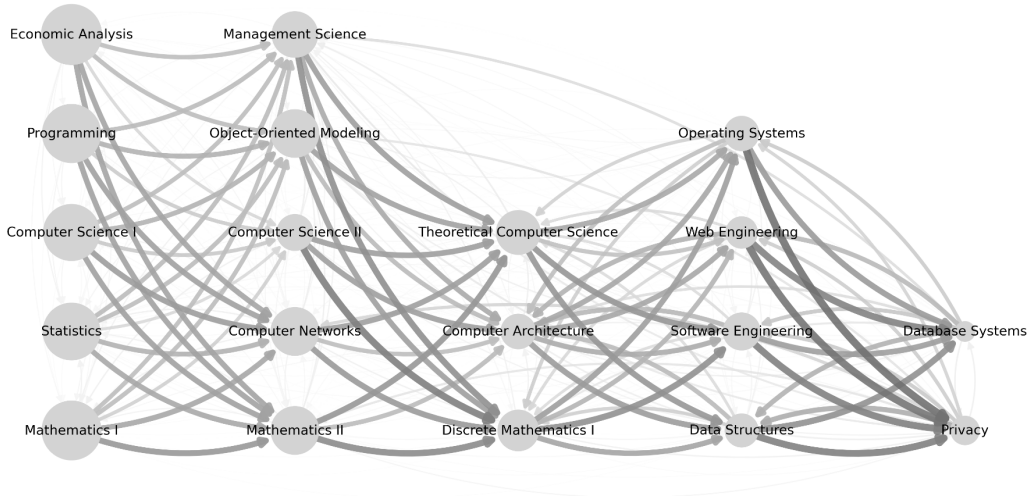


Figure 1: Student Flow based Curriculum Graph G_f

flow graph G_f as follows. First, the courses that have not been passed before are selected. Given the remaining workload, we draw courses based on the courses already passed using the graph G_f until the workload is reached. For this purpose, we go through the passed courses in random order. For each passed course, we draw one course from the not yet passed neighboring courses from V_f , which are connected to the current course via edges from E_f . The associated transition rates W_f are used in the drawing process as transition probabilities. If a course is selected, it can no longer be drawn over other courses.

3.2.3 Pass/Fail Prediction

Grade prediction depends on the quality of the data that is available. Group G^* consists of students who have very low failure rates, as we have shown above. To examine which is a good prediction method for all data in our data set, we want to obtain a subset that is as representative as possible. We consider student groups $G(t)$, which contain all students who attempted all courses from term t . This gives us the cardinalities $|G(1)| = 245$, $|G(2)| = 111$, $|G(3)| = 86$, $|G(4)| = 56$, and $|G(5)| = 50$. To test the accuracy of different methods, we calculate the average accuracy values over all terms t . In addition, we limit ourselves to the labels "passed" and "failed". As a baseline, we use the prediction that always predicts the most frequent class 'passed' (72.2%) and compare it with the following methods. The first method 'Global' draws 'pass' with the average pass rate as probability, independently of course and student. The method 'Course' uses course-specific pass rates and the method 'Student' uses student-specific pass rates. In addition to these simplified methods, we use Naive Bayes and Decision Tree, implemented using scikit-learn [16], based on the promising results in past studies [2]. Balanced accuracy (bACC) defined as

$$\text{bACC} := \frac{1}{2} \left(\frac{\text{TP}}{\text{TP} + \text{FN}} + \frac{\text{TN}}{\text{TN} + \text{FP}} \right),$$

is used as a measure of accuracy, where TP/FP and TN/FN are the true/false positives and true/false negatives, respec-

tively [6]. In addition to predicting new grades, grades must also be predicted when a student repeats a course exam. We have seen that repeating students perform better on average in the second attempt of an exam. If a student fails an exam, the grade of the next try is predicted to be the old grade plus a grade bonus, which equals the average performance gain of 2.622. To fit the grade scale, the grade bonus is rounded up to 3. It is further assumed that the bonus applied from the second to the third attempt does not differ from the bonus applied from the first to the second attempt.

3.2.4 DCT Simulation Assumptions

The following summarizes the assumptions used for the DCT-SIM.

1. All courses can be attended every semester and all students attend courses every semester.
2. Workload and first semester grades are drawn from the data.
3. If a student's drawn workload is 0, we draw a workload from the normal distribution given the mean and the standard deviation of the other student's workloads.
4. If we need to draw workloads without a data basis, we set them to 15.
5. Students do equally better in all courses when they retake an exam (grade bonus = 3).
6. DCT-SIM finishes when every course is finished.

3.3 Policy Changes

Policy changes to the curriculum are a powerful tool to change student behavior. Its use should be all the more careful. Our DCT-SIM offers the ability of an approximate evaluation of these sensitive changes. First, we consider the workload. Since the proposed number of courses in the first semester results in a workload of 27 credits, we examine the effect of changing the workload. We simulate the impact of

increasing the workload by exactly one course, which corresponds to a minimum of 5 credits. Second, we consider the course exams. In the current curriculum, one exam per semester is offered. This is different in other degree programs where there are two exams every two semesters or even two exams every semester. This means that students may be able to continue their studies without delay even if they do not pass the first exam. The policy changes are summarized as follows:

1. 'Workload': 5 credit raise of workload in the first term,
2. 'Exam (2-2-2)': Every exam takes place twice every term,
3. 'Exam (2-0-2)': Every exam takes place twice only every two terms.

3.4 Evaluation Metrics

As in Fiallos et al. [9], we use the mean DCT to compare two DCT distributions to benchmark our results. The authors additionally used a Mann-Whitney U test to evaluate the statistical significance of the simulated DCT distribution. The test can support the alternative hypothesis that two samples correspond to different probability distributions rejecting the null hypothesis that the corresponding distributions are equal. Since the p -value is defined as the conditioned probability of the observed statistic conditioned on the null hypothesis, a high p -value fails to reject the null hypothesis but does not necessarily accept it [24]. Therefore, we will instead simulate individuals and thus introduce a DCT based L^1 error measure using the ground truth DCT as

$$E_{\text{DCT}} := \frac{1}{|S|} \sum_{i=1}^{|S|} \left(\frac{1}{|G^*|} \sum_{j=1}^{|G^*|} |\text{DCT}_{\text{simulated}}^j - \text{DCT}_{\text{data}}^j| \right)_i,$$

where $|S|$ corresponds to the number of DCT-SIM runs. In the following, we set $|S| = 100$.

4. RESULTS

4.1 Pass/Fail Prediction and DCT Simulation

Since in the baseline method we always assume label 'pass', the true positives (TP) correspond to the actual pass rate and the false positives (FP) to the actual failure rate across all courses in the degree program. Table 4 shows the pass/fail prediction results of the various methods. For the baseline method the true positive percentage (TP) is high reflecting that the pass rate across all courses in the degree program is 0.722. The balanced accuracy (bACC) is 0.5. We note that the accuracies of the Global (0.555) and Course (0.553) methods are not significantly greater than the baseline. On the other hand, the Student method (0.608) achieved the best values for true negative and false negative percentages. The Naive Bayes method achieved the highest true positive percentage, while the Decision Tree method had the best overall bACC with 0.648 and the best false positive percentage. Therefore, we used the Decision Tree method in the DCT-SIM.

Figure 2 shows the DCT distributions of the DCT-SIM in light grey and the data in dark grey. The bars indicate the

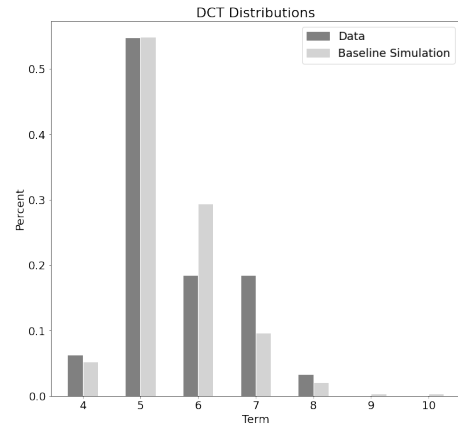


Figure 2: Evaluation of DCT Simulation

average percentage values achieved over a total of 100 simulation runs. The DCT distributions of the simulation and the data reveal similar values in terms 4 and 5. In the other terms, when the data of the G^* group are sparser, since most of the students already completed all courses, the accuracy of the DCT-SIM also decreases noticeably. However, there is a small error of 0.087 between the mean values of the DCT distributions with a p -value of 0.897, which is comparable to Fiallos et al. [9]. According to our DCT based L^1 error measure, we reach a value of $E_{\text{DCT}} = 0.926$. This value corresponds to an average individual DCT error of 0.926 semesters.

4.2 Policy Changes

In the DCT simulations of the policy changes, for the sake of clarity, we have chosen a cumulative representation of the DCT distributions in Figure 3. The two graphs in gray correspond to the distributions of the data and baseline DCT-SIM as shown in Figure 2. We see that the workload change appears to be very effective, as the graph appears to be consistently above that of the baseline DCT-SIM. It turns out to be an improvement in the DCT means of -0.232 . Thus, this change leads to a shortening of the DCT. The minor improvement of -0.035 by policy change 'Exam (2-2-2)' is only noticeable from term 6. In contrast, the policy change 'Exam (2-0-2)' leads to a global increase of the DCT of 0.591.

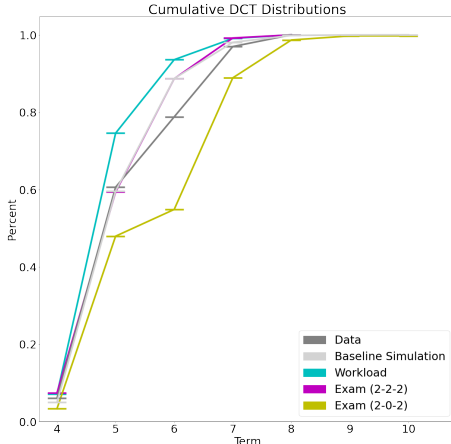
5. DISCUSSION AND LIMITATIONS

We were able to show that student-dependent pass/fail prediction using the groups $G(t)$ performs better than the assumption of a global pass rate or a course-specific pass rate. Overall, it is worthwhile to use the existing student performance data to increase accuracy.

Regarding the DCT-SIM, research in student flow simulation lacks good evaluation methods. Evaluation methods on the individual student level are missing, instead global DCT or dropout distributions are compared with the ground truth. This is due to the lack of use of labeled data and the resulting loss of assignability of students to their simulated counterparts. Using our generalizing approach, we were able to obtain comparable mean DCT errors on a global scale. In addition, through the intensive use of data, we were able to

Table 4: Pass/Fail Prediction Methods Scores with 70:30 Train-Test Split

Score \ Methods	Baseline	Global	Course	Student	NB	DT
TP	0.722	0.574	0.585	0.643	0.701	0.645
FP	0.278	0.223	0.226	0.182	0.143	0.124
TN	0.000	0.055	0.052	0.096	0.068	0.088
FN	0.000	0.148	0.137	0.079	0.087	0.143
bACC	0.500	0.555	0.553	0.608	0.617	0.648

**Figure 3: Evaluation of Simulated Policy Changes**

evaluate our DCT-SIM individually for each student from group G^* . The error measure E_{DCT} has shown that the evaluation with an average individual misfit of 0.926 can be quite high without having a globally large effect. This results in the necessity to specify the goodness of fit of a DCT-SIM not only on a global level in the form of a mean DCT. We believe that a good DCT-SIM should be able to simulate each student as accurately as possible. Otherwise, a practical implementation of changes based on the observed DCT-SIM could lead to unexpected dynamics and thus to unfair conditions for an individual student. Policy changes for a curriculum are a relevant use case for a DCT-SIM because a data-generating process is needed to produce results. For our approach, we could identify two types of changes: Additive policy changes and restrictive policy changes. The former is characterized by changes in the frequency or acceleration in time of exams. For example, an increase in workload conditions the speed at which courses can be attended. The latter type, under which the 'Exam (2-0-2)' change falls, directly restricts the frequency of exams in every second term. Our method was able to evaluate changes in the first category very well. The policy changes 'Workload' and 'Exam (2-2-2)' show plausible results. In particular, 'Exam (2-2-2)' equates to a global increase in pass rates in our setting, due to the assumption of a grade bonus when repeating an exam. Since we have shown that students from group G^* have very high pass rates, it makes sense that the effect is small for this group.

The 'Exam (2-0-2)' change appears to be leading to a global extension of DCT. We have seen that the policy change 'Exam (2-2-2)' behaves similarly to the baseline simulation. Therefore, we conclude that the policy change 'Exam (2-0-

2)' also behaves similarly to the policy change 'Exam (1-0-1)'. We point out the step-shaped graph in Figure 3. Looking at the number of courses per semester in Figure 1 and the average semester workloads of group G^* in Table 3, we see that in semesters 1, 3 and 5 the average workload difference from the recommended workload is higher than in semesters 2 and 4. Thus, the case occurs where a student has completed all subjects from semesters 2 and 4 in semester 6, but is missing subjects from semesters 1, 3 and 5 that cannot be completed until semester 7 or even 9. As a result, few students finish in 6 or 8 semesters, as seen in Figure 3. This shows the still existing rigidity of our workload approach. We believe that predicting workload as a function of courses, grades, and exam attempts, rather than deriving it directly from the student being simulated, may lead to a more robust DCT-SIM with respect to policy changes.

Awareness regarding algorithmic bias and fairness is emerging in the field of Educational Data Mining. Especially when a model gets implemented in practice so that students are affected in their learning environment, an identification of bias and fairness of the data and methods used as well as their resulting actions have to be assured [4, 11]. In addition to the mentioned limitations of our method, we are aware of the performance bias (i.e. we mainly consider successful students) in the data used and would like to remove it in the future by expanding and balancing our data set.

In terms of computational power, we observe a time consumption for one non parallelized DCT-SIM of student group G^* : $|G^*|it \cdot 0.714 \text{ s/it} = 33 \text{ it} \cdot 0.714 \text{ s/it} = 23.571 \text{ s}$, using an Intel(R) Core(TM) i5-6200U CPU @ 2.3GHz processor.

With respect to the DCT mean, we were able to obtain a comparatively well-fitting DCT distribution from our DCT-SIM in which only student data were processed. The dependence on the data offered us the ability to construct a student flow based curriculum graph in a maximally flexible study plan, perform an individual evaluation of the DCT-SIM using the E_{DCT} error measure and investigate and evaluate policy changes in a highly flexible curriculum. In this sense, we were able to generalize existing approaches. In order to implement our approach for university practitioners in the future, we will conduct further research. The first priority is the extension of the data set as well as achieving high individual accuracy. An ethical analysis with the involvement of student and teacher representatives is planned.

6. ACKNOWLEDGMENTS

We thank Agathe Merceron for her valuable feedback. The work is supported by the Ministry of Culture and Science of the state of NRW, Germany through the project KI.edu:NRW.

7. REFERENCES

- [1] P. R. Aldrich. The curriculum prerequisite network: Modeling the curriculum as a complex system. *Biochemistry and Molecular Biology Education*, 43(3):168–180, 2015.
- [2] R. Asif, A. Merceron, S. A. Ali, and N. G. Haider. Analyzing undergraduate students’ performance using educational data mining. *Computers & Education*, 113:177–194, 2017.
- [3] M. Backenköhler, F. Scherzinger, A. Singla, and V. Wolf. Data-driven approach towards a personalized curriculum. In *11th International Conference on Educational Data Mining*, pages 246–251, 2018.
- [4] R. S. Baker and A. Hawn. Algorithmic bias in education. *International Journal of Artificial Intelligence in Education*, pages 1–41, 2021.
- [5] J. Banks. *Handbook of simulation: principles, methodology, advances, applications, and practice*. John Wiley & Sons, 1998.
- [6] M. Bekkar, H. K. Djemaa, and T. A. Alitouche. Evaluation measures for models assessment over imbalanced data sets. *Journal of Information Engineering and Applications*, 3:27–38, 2013.
- [7] A. Bogarín, R. Cerezo, and C. Romero. A survey on educational process mining. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(1):e1230, 2018.
- [8] E. Caro, C. González, and J. M. Mira. Student academic performance stochastic simulator based on the monte carlo method. *Computers & Education*, 76:42–54, 2014.
- [9] A. Fiallos and X. Ochoa. Discrete event simulation for student flow in academic study periods. In *2017 Twelfth Latin American Conference on Learning Technologies (LACLO)*, pages 1–7. IEEE, 2017.
- [10] A. Hagberg, P. Swart, and D. S Chult. Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008.
- [11] R. F. Kizilcec and H. Lee. Algorithmic fairness in education. *arXiv preprint arXiv:2007.05443*, 2020.
- [12] E. Kyndt, I. Berghmans, F. Dochy, and L. Bulckens. ‘time is not enough.’workload in higher education: a student perspective. *Higher Education Research & Development*, 33(4):684–698, 2014.
- [13] S. Mansmann and M. H. Schöll. Decision support system for managing educational capacity utilization. *IEEE Transactions on Education*, 50(2):143–150, 2007.
- [14] R. Molontay, N. Horváth, J. Bergmann, D. Szekrényes, and M. Szabó. Characterizing curriculum prerequisite networks by a student flow approach. *IEEE Transactions on Learning Technologies*, 13(3):491–501, 2020.
- [15] M. Pechenizkiy, N. Trcka, P. De Bra, and P. Toledo. Currim : Curriculum mining. In *Proceedings of the 5th International Conference on Educational Data Mining, EDM 2012*, pages 216–217. International Educational Data Mining Society (IEDMS), 2012.
- [16] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [17] M. Raji, J. Duggan, B. DeCotes, J. Huang, and B. V. Zanden. Modeling and visualizing student flow. *IEEE Transactions on Big Data*, 7(3):510–523, 2021.
- [18] C. Romero and S. Ventura. Educational data mining and learning analytics: An updated survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 10(3):e1355, 2020.
- [19] R. M. Saltzman and T. M. Roeder. Simulating student flow through a college of business for policy and structural change analysis. *Journal of the Operational Research Society*, 63(4):511–523, 2012.
- [20] A. Schellekens, F. Paas, A. Verbraeck, and J. J. van Merriënboer. Designing a flexible approach for higher professional education by means of simulation modelling. *Journal of the Operational Research Society*, 61(2):202–210, 2010.
- [21] A. Slim, G. L. Heileman, J. Kozlick, and C. T. Abdallah. Employing markov networks on curriculum graphs to predict student performance. In *2014 13th International Conference on Machine Learning and Applications*, pages 415–418. IEEE, 2014.
- [22] A. Slim, G. L. Heileman, J. Kozlick, and C. T. Abdallah. Predicting student success based on prior performance. In *2014 IEEE symposium on computational intelligence and data mining (CIDM)*, pages 410–415. IEEE, 2014.
- [23] P. Voigt and A. Von dem Bussche. The eu general data protection regulation (gdpr). *A Practical Guide, 1st Ed., Cham: Springer International Publishing*, 10(3152676):10–5555, 2017.
- [24] R. L. Wasserstein and N. A. Lazar. The asa statement on p-values: context, process, and purpose, 2016.
- [25] J. Wigdahl, G. L. Heileman, A. Slim, and C. T. Abdallah. Curricular efficiency: What role does it play in student success? In *2014 ASEE Annual Conference & Exposition*. ASEE Conferences, 2014.

Modeling One-on-one Online Tutoring Discourse using an Accountable Talk Framework

Renu Balyan
SUNY College at Old Westbury,
New York, USA
balyanr@oldwestbury.edu

Tracy Arner
Arizona State University,
Arizona, USA
tarn@asu.edu

Karen Taylor
Arizona State University,
Arizona, USA
karnetaylor.sb@gmail.com

Jinnie Shin
University of Florida,
Florida, USA
jinnie.shin@coe.ufl.edu

Michelle Banawan
Arizona State University,
Arizona, USA
mbanawan@asu.edu

Walter L. Leite
University of Florida,
Florida, USA
walter.leite@coe.ufl.edu

Danielle S. McNamara
Arizona State University,
Arizona, USA
dsmcnamara1@gmail.com

ABSTRACT

The National Council of Teachers of Mathematics (NCTM) has been emphasizing the importance of teachers' pedagogical communication as part of mathematical teaching and learning for decades. Specifically, NCTM has provided guidance on how teachers can foster mathematical communication that positively impacts student learning. A teacher may have different academic goals towards what needs to be achieved in a classroom, which require a variety of discourse-based tools that allow students to engage fully in mathematical thinking and reasoning. Accountable or academically productive talk is one such approach for classroom discourse that may ensure that the discussions are coherent, purposeful and productive. This paper discusses the use of a transformer model for classifying classroom talk moves based on the accountable talk framework. We investigate the extent to which the classroom Accountable Talk framework can be successfully applied to one-on-one online mathematics tutoring environments. We further propose a framework adapted from Accountable Talk, but more specifically aligned to one-on-one online tutoring. The model performance for the proposed framework is evaluated and compared with a small sample of expert coding. The results obtained from the proposed framework for one-on-one tutoring are promising and improve classification performance of the talk moves for our dataset.

Keywords

accountable talk framework, classroom discourse, one-on-one online tutoring, transfer learning.

R. Balyan, T. Arner, K. Taylor, J. Shin, M. Banawan, W. Leite, and D. McNamara. Modeling one-on-one online tutoring discourse using an accountable talk framework. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 477–483, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852936>

1. INTRODUCTION

Productive classroom discourse is positively associated with student learning [6,7,13,15] across multiple content areas, including reading comprehension [18], academic vocabulary learning [9] development of collaborative reasoning [14], persuasive writing performance [1] historical reasoning [26], scientific argumentation [5], and mathematical reasoning [13]. Additionally, academically rigorous classroom discussions are explicitly promoted in mathematical pedagogy [3,19] and in the Common Core State Standards [20] that guide teachers' instructional practices.

The oft-cited, common pattern of classroom talk, Initiation-Response-Evaluation (IRE) [11] emerged in early research investigating the forms and functions of classroom talk [12]. This minimal unit of interactional exchange includes a teacher's initiation, then a student's response, followed by the teacher's evaluation of the response. The IRE pattern is most commonly noted in teacher-led lessons (i.e., direct instruction) which tends to be the default in many classrooms. The IRE pattern of classroom talk demonstrates a transmission style of instruction such that the teacher is in a position of authority and controls the content of the discourse as well as who engages in it [4]. Indeed, this pattern of monologic classroom discourse is still common in many classrooms despite research demonstrating the benefit of dialogic and reasoning-based discourse [6,15,17].

Dialogic discourse, on the other hand, aligns with Vygotsky's [25] sociocultural theory stipulating that learning is more likely to occur when thinking is socialized, particularly within the range of a student's ability when provided with appropriate guidance (i.e., Zone of Proximal Development; ZPD). Interactions between teachers and students serve as a scaffold for knowledge acquisition during instruction [21,24]. A hallmark of this type of scaffolded, dialogic classroom discourse is the *Accountable Talk framework* [15].

2. ACCOUNTABLE TALK FRAMEWORK

The Accountable Talk Framework divides teacher and student discourse into three broad categories (see Table 1 for definitions and respective examples): accountability to the community, accountability to accepted standards of reasoning, and accountability to knowledge [15]. Talk that is accountable to the community is characterized as cutting across disciplines and “attends seriously to and

builds on the ideas of others” [15, p. 286]. Accountability to standards of reasoning involves making logical and reasonable connections, explanations, and conclusions. Accountability to knowledge emphasizes sensemaking based on facts and authoritative knowledge. The latter categories of discourse practices promoting accountability to standards of reasoning and knowledge are generally more discipline-specific.

Table 1. Accountable Talk framework teacher and student talk moves definitions and examples

Talk Moves Definition and Example(s)	
Teacher Talk Move	
<i>Learning Community</i>	
Keeping Together	Keeping everyone together; prompting students to be active listeners and orienting students to each other. For example, turn to the student sitting next to you; what do you think; What did X just say the equation was?
Relating Students	Getting students to relate to other students’ idea; prompting students to respond to what a classmate said. For example, would someone like to add on to what X said; do you agree with Y that the answer is; does anyone understand how Z solved this problem?
Restating	Repeating all or in part of what a student said. For example, Student: the factors, Teacher: factors
<i>Content Knowledge</i>	
Accuracy	Pressing for accuracy; prompting students to make a mathematical contribution or use mathematical language. For example, can you give an example of X; What’s another word for that?
<i>Rigorous Thinking</i>	
Revoicing	Repeating what a student said but changing the wording or adding to it; using at least a key mathematical word or idea from what the student said. For example, Student: it is x squared; Teacher: so, instead of a cube it will be squared.
Reasoning	Pressing for reasoning; prompting students to explain or provide evidence. For example, can you explain why you think this is the answer; why would you add two and not three?
Student Talk Move	
<i>Learning Community</i>	
Relate to another student	Comment on or ask questions about another student’s idea or use another student’s idea to form your own basis. For example, I got the same answer as X; I was about to say what Y just said.
Asking for more information	Ask for help in case confused or request more information about a math topic. For example, can you please explain this again; I did not understand this; Is this multiplication or division?
<i>Content Knowledge</i>	
Making a claim	Make a factual statement about a mathematical concept; list a step to arrive at an answer; make a mathematical claim. For example, I got this answer by dividing the two numbers; X is the profit.
<i>Rigorous Thinking</i>	
Providing evidence or explanation	Provide the evidence or explanation for the mathematical claim or explain their thinking. For example, you cannot divide the number by zero as that will result in an infinite number; four multiplied by five gives me twenty.

Prior research evaluating implementation and benefit of discourse using dialogic frameworks was conducted by transcribing and hand coding hours and hours of classroom audio or video recordings [22]. This method is labor intensive and time consuming thus making it difficult to provide teachers with feedback on their implementations of academic discourse. Suresh and colleagues [22] used automatic speech recognition and machine learning to evaluate teacher-student interactions using the Accountable Talk framework. The machine learning models trained as part of their TalkBack application showed promise as a medium to provide teachers with important feedback regarding their pedagogical practices. However, this work has only been applied to traditional classroom instruction. Therefore, less is known about the application of the Accountable Talk framework in discourse occurring outside of the traditional classroom. The recent shift to hybrid or online learning necessitated by the COVID-19 pandemic, provides a rich opportunity to apply machine learning models to teacher-student interaction in an online environment. The capability to evaluate academic discourse and provide feedback to teachers that

may lead to improved student learning is particularly important given the concern over learning gaps being exacerbated by the pandemic [2].

The present study builds on this work using the machine learning models developed by Suresh and colleagues to detect evidence of the Accountable Talk framework [15] in teacher-student interactions in online, one-on-one tutoring. For the purpose of this study, we adopted the same talk moves outlined in Suresh [22]. We further refined our analysis by including moves that would be included in a one-on-one tutoring interaction and excluded those that could only occur in traditional, whole class settings. Specifically, talk moves related to accountability to the community were removed from data for the model to be re-trained for the proposed framework as they are focused on ensuring learners have a shared focus and on developing collegial interactions between students. Since there was only one student and one teacher in each tutoring session, no utterances exemplifying these two talk moves were present in our data.

3. METHOD

3.1 Participants

The student population was recruited for online tutoring in summer and fall of 2019 from four high schools in the Broward school district. Participants included 40 students who did not pass the Algebra 1 course or were not successful on the end of course (EOC) exam. Tutoring was provided during school hours on school grounds in the summer or fall semester. Sessions were planned for up to 10 hours, but the number of hours varied from 1 to 20 with a mean of 5 hours of tutoring. Tutoring was conducted by credentialed math teachers with at least two years of experience in the Florida system to ensure that they were familiar with the targeted curriculum standards. Tutoring sessions were conducted online using Study Edge’s GoBoard video conferencing system which supports shared note-taking (pen-casts). During the online tutoring sessions, both the student’s and the tutor’s computer screens were recorded, which included an audio and video recording of their conversation. The audio recordings were later transcribed to obtain the student-teacher discourse during the tutoring session. This study was approved by the Institutional Review Board and all participants in this study consented to the use of their data in accordance with APA guidelines.

3.2 Data

Three different data sets were used in this study. The first set was the talk moves labeled data available from prior research. The second data set was a subset of the first set that was filtered for talk moves. The second data set was more aligned with the one-on-one tutoring discourse. The third data set consisted of the unlabeled data containing teacher and student utterances from one-on-one tutoring discourse. Examples of teacher and student talk moves from the data sets discussed below are shown in Table 2.

Table 2. Talk Moves examples (Teacher Talk Moves – 1: none; 2: Keeping everyone together, 3: getting students to relate, 5: re-voicing, 6: Press for accuracy; Student Talk Moves – 1: none, 2: relating to another student, 3: asking for more information, 4: making a claim, 5: providing evidence/explaining reasoning)

Speaker	Sentence	Teacher Tag	Student Tag
Teacher	Who can help her notice where she went wrong	3	nan
Teacher	<<student1 name>>	2	nan
Student	She kept the six	nan	2
Teacher	<<student2 name>>	2	nan
Student	She put the eight so everything after the decimal number that you need changed	nan	5
Teacher	You underline you look at an arrow then what	6	nan
Student	Whats that one negative 51	nan	3
Teacher	Mixed numbers you were right	5	nan
Student	Its like one and one half or one and three thirds or something like that	nan	4
Teacher	All right I see some disagreements here	1	nan
Student	I was waiting	nan	1

3.2.1 Data Set 1

The Data Set 1 included annotated transcripts of classroom discourse in mathematics collected in public schools in the US and are, therefore, aligned with our Algebra tutoring data set. This talk moves data was obtained from [23], which included entire lessons and short excerpts from lessons. The data set consisted of 230,778 utterances (172,309 teacher utterances and 58,469 student utterances) from 559 lesson transcripts.

3.2.2 Data Set 2

The full data set (Data Set 1) was filtered for talk moves applicable to one-on-one tutoring discourse from the original data set [23]. The filtered dataset consisted of 199,123 utterances (147,145 teacher utterances and 51,978 student utterances) from 558 lesson transcripts.

3.2.3 Data Set 3

The tutoring data used to classify the talk moves consists of 87,100 utterances (62,370 teacher and 24,730 student utterances). The final dataset consisted of transcripts from 130 one-on-one tutoring sessions between 25 teachers and 39 students.

3.2.4 Talk Moves data set creation via expert coding

Expert coding of the data was conducted in two phases: training and data coding. Training was conducted using the Data Set 1 consisting of transcripts for individual student-teacher conversations. The transcripts in this original data were coded for six mutually exclusive teacher talk moves, and five student talk moves that were adapted from the Accountable Talk framework. The teacher or student utterances that did not contain a talk move were labeled as “None-1”. Strings of utterances by the teachers that were aimed at direct instruction were considered “None-1” as they were not aimed at fulfilling one of the six accountable talk moves outlined in the framework. Four undergraduate research assistants were trained to code a sample of each teacher and student ‘talk move’.

The first step in training the coders involved acquainting them with each of the types of talk moves explicated in the Accountable Talk framework [15]. Research assistants met with a researcher and discussed examples of each type of Talk Move for both teachers and students. Importantly, coders learned to determine which utterances were part of direct instruction or not consequential (e.g., classroom management utterances such as “get out your textbook”) to eliciting student responses; these are coded as “1-None”. The talk moves being classified in this framework consist of turns of an interaction between student(s) and teachers with the goal of knowledge construction. Following the initial group training, each coder was given one of two practice sets created from the large data set provided by [23]. Equal numbers of each type of move were included in the data set such that coders had 100 target sentences for each of the six mutually exclusive teacher talk moves and five of the student talk moves. The teacher and student talk moves were separated (i.e., one practice sheet for teachers and one for students) and randomized so that no patterns could be detected. The interrater reliability (weighted kappa) for the first pair of coders for teacher practice set A was 0.44 and 0.61 for student practice set A. The interrater reliability (weighted kappa) for the second pair of coders was 0.65 for both teacher and student practice set B. The moderately low interrater reliability suggested that either coders were not quite familiar enough with the codes or that the utterances were too difficult to code when taken out of the context of the discourse. Using the student utterance was necessary to accurately determine the context, particularly for the restating and revoicing teacher talk moves. Additionally, multiple utterances may be included in one sentence and there may be multiple sentences included within a turn. Therefore,

all four coders were given a second practice data set (revised version) that included utterances in context. For example, coders received spans of sentences that included both the teacher utterances and the student utterances. Prior to starting on the new practice data set, coders were provided with a short training refresher video. The interrater reliability for the student tags in the second practice set ranged from 0.42 to 0.74 with a weighted kappa of 0.60 for all raters. The interrater reliability for the teacher tags in the second practice set ranged from 0.52 to 0.71 with a weighted kappa of 0.60 for all raters. Weighted kappa ranging from 0.41 to 0.60 suggest moderate agreement [8]. Following the training sets, the two coders with the highest interrater reliability were provided with a subset of Data Set 3 to code, containing 570 teacher utterances and 183 student utterances. The interrater reliability for the experimental data was 0.90 weighted kappa for teacher utterances and 0.95 weighted kappa for student utterances. Weighted kappa over 0.81 suggest nearly perfect agreement [8].

3.2.5 Model architecture

Because our data (Data Set 3) was not labeled/coded for talk moves initially, we used the labeled data (Data Sets 1 and 2) available from prior research [23] and the filtered data set for initial model development. The labeled data was split into training (75%), validation (5%) and test (20%) sets to verify that the prior research results could be replicated. We used the RoBERTa-base transformer model [10], a model pretrained with Masked Language Modeling (MLM) using five data sets including BookCorpus - a dataset consisting of 11,038 unpublished books; English Wikipedia; CC-News - a dataset containing 63 million English news articles; OpenWebText - an open-source recreation of the WebText dataset used to train GPT-2; and Stories - a dataset containing a subset of CommonCrawl data, to train our model. RoBERTa allows the model to learn a bidirectional representation of a sentence.

The models were trained using sentences that were preprocessed into *turns*, where each turn constituted a student utterance followed immediately by a teacher utterance for the teacher talk moves model. Similarly, for student talk moves, each turn consisted of a teacher utterance immediately followed by a student utterance. A teacher or a student turn could also include multiple utterances. There were very few student-student pair utterances as compared to the original/prior data (classroom setting) because our data constituted one-on-one tutoring transcripts. Both the student and teacher utterances were required to set the context particularly for the *restate* and *revoice* teacher talk moves, and *make a claim* and *provide evidence or reasoning* student talk moves. The input sentences were also cleaned of any punctuation and converted to lowercase. The pretrained model from the Hugging Face library was used and the model was trained using a Tesla P100 GPU. Code was implemented using TensorFlow framework with Python version 3.7. The `batch_encode_plus` method was used for tokenizing and encoding the pair of sentence sequences and AdamW was the optimizer. The hyperparameters used for training and tuning the model included a learning rate of $2e-5$, and number of epochs and batch size as 4. The talk moves were used as the dependent variables.

3.2.6 Analytic framework

The study was performed as a set of five experiments. First, a transformer model was trained and tested using Data Set 1 [23], the model trained using Data Set 1 was then used to classify our unlabeled tutoring data (Data Set 3) for both student and teacher talk moves. The second experiment was performed to compare the consistency between the distributions of the talk moves for Data Set 1

and the predicted talk moves for Data Set 3. In the third experiment, after determining the consistency between the talk moves distribution of the two data sets (Data Set 1 and Data Set 3), the prior data (i.e., Data Set 1) was filtered for talk moves that are more applicable to one-on-one online tutoring. The teacher talk moves *keeping everyone together* and *getting students to relate* were removed as these talk moves are more relevant to classroom teaching rather than one-on-one online tutoring. Similarly, the utterances labeled as student talk move *relating to another student* were also removed. Prior research data, filtered for one-on-one online tutoring aligned talk moves (Data Set 2), was used to re-train and re-test the model. The re-trained model was used to classify our unlabeled data (Data Set 3) again for both the teacher and student one-on-one tutoring related talk moves. The distribution of the talk moves for the filtered datasets was compared again to confirm whether the newly trained model classified the talk moves consistently with the prior research data that was labeled in the fourth experiment. The fifth experiment evaluated the performance of the model that was trained using talk moves applicable to one-on-one tutoring. The predicted talk moves for Data Set 3 were compared with the expert coded talk moves for a small sample of Data Set 3.

4. RESULTS

4.1 Model Performance for Data Set 1: Original Framework

The first set of experiments included training a transformer model using the Data Set 1 (training set) and classifying the teacher and student talk moves on the test set for the talk moves defined in the original accountable talk framework. The confusion matrix for the teacher and student talk moves for Data Set 1 (test set) are shown in Figure 1 (a and b), respectively. The micro F1 and macro F1 scores obtained for the teacher test set were 0.89 and 0.79, respectively. The Matthew correlation coefficient (MCC) was 0.79. The F1 scores for the student talk moves were lower than the teacher talk moves and were 0.80 (micro F1) and 0.76 (macro F1), and the MCC was 0.71. The precision, recall, and F scores of the talk moves for both teachers and students were also computed. The teacher talk moves *getting students to relate* and *revoicing* had the lowest F scores, 0.68 and 0.67 respectively. The teacher talk moves *press for reasoning*, *press for accuracy*, and *restating* had higher and comparable F scores (0.80-0.84). The student talk move *relating to another student* performed the worst with an F score of 0.58 and other student talk moves had higher and comparable F scores (0.76-0.79).

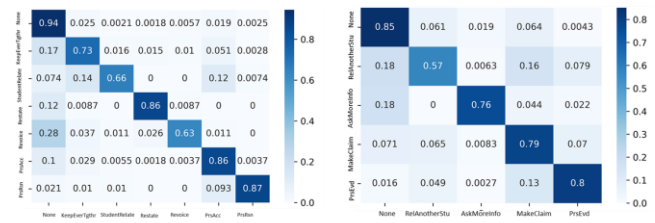


Figure 1. (a) Teacher Test set Talk Moves

Figure 1. (b) Student Test set Talk Moves

Figure 1. The teacher and student talk move accuracy and labels: None (*no talk move*), KeepEverTgthr (*keeping everyone together*), StudentRelate (*getting students to relate to other students' ideas*), Restate (*restating*), Revoice (*revoicing*), PrsAcc (*press for accuracy*), and PrsRsn (*press for reasoning*), RelAnotherStu (*relating to another student*), AskMoreInfo (*asking for more information*), MakeClaim (*making a claim*), and PrsEvd (*providing evidence or explaining reasoning*).

4.2 Talk Moves Distribution Comparison (Data Set 1 vs. Data Set 3): Original Framework

The second experiment included comparing the talk moves distribution of the original data (Data Set 1) and the predictions for our data (Data Set 3) obtained using the model trained in the first experiment. The distribution of the teacher and the student talk moves across the two datasets was found to be comparable and consistent with slight variations. Both datasets found 67.42-76.03% of utterances did not contain a teacher talk move. The remaining teacher talk moves in order of frequency were, *pressing for accuracy* (10.59-13.07%), *keeping everyone together* (9.44-12.97%), and *revoicing* (2.15-2.27%). *Getting students to relate to another student's idea* was the least classified (0.14%) teacher talk move in our dataset (Data Set 3). The remaining teacher talk moves shown in Figure 2 (a and b) had similar distributions.

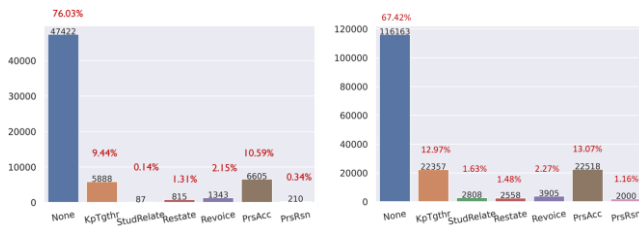


Figure 2. (a) Teacher Talk Moves Classification for Data Set 3 Figure 2. (b) Teacher Talk Moves Classification for Data Set 1

Figure 2. The teacher talk move distribution and labels for Data sets 3 and 1: None (*no talk move*), KpTgthr (*keeping everyone together*), StudRelate (*getting students to relate to other students' ideas*), Restate (*restating*), Revoice (*revoicing*), PrsAcc (*press for accuracy*), and PrsRsn (*press for reasoning*).

The predicted student talk moves also had consistent distribution with the prior labeled data (Data Set 1). Following *not a talk move*, the student talk move *making a claim* was the second most abundant with 28.25-30.66% utterances falling into that category. *Asking for more information* was the least observed talk move in both the data sets (see Figure 3 (a and b)).



Figure 3. (a) Student Talk Moves Classification for Data Set 3 Figure 3. (b) Student Talk Moves Classification for Data Set 1

Figure 3. The student talk move distribution labels for Data sets 3 and 1 are: None (*no talk move*), RelateStu (*relating to another student*), MoreInfo (*asking for more information*), claim (*making a claim*), and ProEvidence (*providing evidence or explaining reasoning*).

The consistent distributions of talk moves across the two datasets combined with very few utterances classified in the talk moves categories of *getting students to relate to another student's idea* (teachers) and *relate to another student* (students) suggest that the classification model trained on Data Set 1 can be used for Data Set 3. The two minimally classified talk moves are not directly applicable or aligned to one-on-one tutoring thus increasing our confidence that the classification model can be used for our one-on-one tutoring dataset with minor updates to the model.

4.3 Model Performance for Data Set 2: Proposed Framework

Experiment 3 was similar to the first experiment except the transformer model was trained using prior data for talk moves that are more aligned and applicable to one-on-one tutoring (i.e., Data Set 2) rather than the original set (Data Set 1) of talk moves. Figure 4 (a and b) shows the confusion matrix for the teacher and student talk moves for the test data from Data Set 2, respectively. The micro F1 for the teacher test set improved from 0.89 to 0.94 and macro F1 score increased from 0.79 to 0.83, respectively. The Matthew correlation coefficient (MCC) was higher (0.79 vs. 0.83). The F1 scores for the student dataset were lower than the teacher talk moves but improved from 0.80 to 0.86 (micro) and 0.76 to 0.82 (macro), and the MCC is higher (0.71 vs 0.78). The precision, recall, and F scores for each talk move, for both teachers and students, are also computed. The teacher talk move *revoicing* had the lowest F score, 0.67, which aligns with the original accountable talk framework. The student talk move *asking for more information* performed the worst with an F score of 0.76.

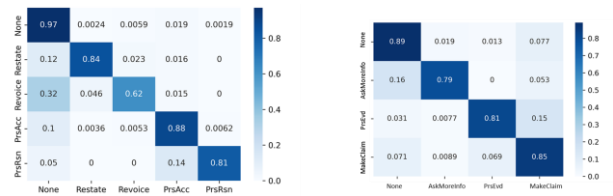


Figure 4.a. Teacher Test set Talk Moves – Proposed Framework Figure 4.b. Student Test set Talk Moves – Proposed Framework

Figure 4. The teacher and student talk move labels for the proposed framework are: None (*no talk move*), Restate (*restating*), Revoice (*revoicing*), PrsAcc (*press for accuracy*), and PrsRsn (*press for reasoning*) and the student talk moves in Figure 4(b) are: None (*no talk move*), AskMoreInfo (*asking for more information*), PrsEvd (*providing evidence or explaining reasoning*) and MakeClaim (*making a claim*).

4.4 Talk Moves Distribution Comparison (Data Set 2 vs. Data Set 3): Proposed Framework

This experiment is comparable to the second experiment except that the distributions of talk moves are for the data aligned to one-on-one tutoring instead of the talk moves from the original accountable talk framework [22,23]. The teacher talk moves distributions for both Data Set 2 and the predicted talk moves for Data set 3 using the model trained in the third experiment were found to be consistent with *press for reasoning* having the lowest frequency (0.38% and 1.36%) and *press for accuracy* as the second highest frequency (11.84% and 15.3%) next to *no talk move*.

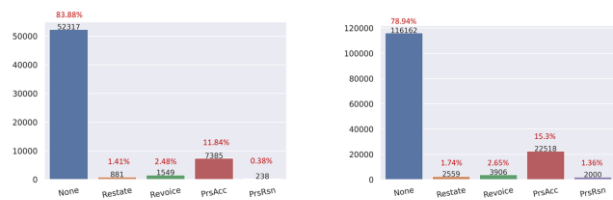


Figure 5. (a) Teacher Talk Moves Predictions for Data Set 3 – Proposed Framework Figure 5. (b) Teacher Talk Moves Classification for Data Set 2 – proposed Framework

The distributions for both Data Set 2 and the predictions of student talk moves for Data Set 3 were similar except for the *providing evidence* talk move (see Figure 6 (a and b)). Very few (3.39% utterances) were classified as *providing evidence* talk moves for

Data Set 3, whereas the original labeled filtered dataset had a higher percentage (~15%) of talk moves belonging to this category.

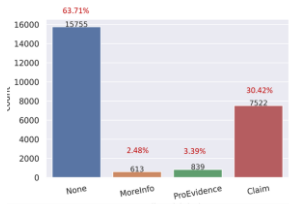


Figure 6. (a) Student Talk Moves Predictions for Data Set 3 – Proposed Framework

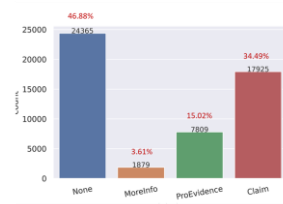


Figure 6. (b) Student Talk Moves Classification for Data Set 2 – proposed Framework

4.5 Model Performance for Data Set 3: Proposed Framework

The student and teacher talk move predictions using the proposed framework model for Data Set 3 when compared with the expert coding achieved 0.88 accuracy for teacher talk moves and 0.81 for student talk moves. The weighted average and micro precision, recall and F scores were in the range of 0.87-0.88 for teacher talk moves and 0.79-0.83 for the student talk moves. The teacher talk move *pressing for reasoning* had the worst performance and the student talk move *asking for more information* performed the worst of all the talk moves. The confusion matrix for both the talk moves are shown in Figure 7 (a and b). The teacher and student talk moves labels in the Figures 7 (a and b) are: *None; restating; revoicing; pressing for accuracy and pressing for reasoning; and None; asking for more information; providing evidence or explaining reasoning; and making a claim for (1-5) and (1-4) respectively.*

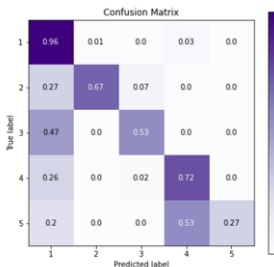


Figure 7. (a) Teacher Talk Moves Classification Predicted vs. Expert Coding – proposed Framework

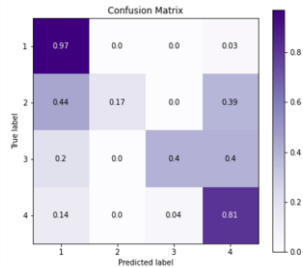


Figure 7. (b) Student Talk Moves Classification Predicted vs. Expert Coding – proposed Framework

5. DISCUSSION

The Accountable Talk framework has been used to classify teacher and student talk moves as a method of providing feedback and guidance to teachers on the discourse strategies they use in their classrooms. Until recently, this type of discourse analysis has been cumbersome and labor intensive. Furthermore, it has been restricted to face-to-face or in-person classrooms mostly. The present work extends the work of Suresh [22,23] in the application of deep learning to a transcribed data set of student and teacher utterances gathered from an online, one-on-one algebra tutoring system. The data set provided by Suresh [23] was used to train our model which was then applied to our unlabeled data set. The data available from prior research (Data Set 1) was used to train a transformer model (RoBERTa) and the test set achieved an accuracy higher than that mentioned in the literature on a similar data set for several deep learning models including Long Short Term Memory (LSTM) unit, Bi-LSTM, gated recurrent unit (GRU), and recurrent neural network (RNN). The Bi-LSTM had outperformed all the other models with an F1 measure of 65% [23]. However, our model RoBERTa achieved an accuracy of 0.89 and 0.94 for the original framework and the proposed framework respectively for the teacher talk

moves. The student talk move model accuracy for the original framework and the proposed framework are 0.80 and 0.86 respectively. The talk move distributions for both teacher and student for our data set (Data Set 3) were also found to be consistent with the prior data (Data Set 1) and the filtered prior data (Data Set 2). In addition, high performance was observed for most of the talk moves for the proposed framework model when applied to our unlabeled data set (Data Set 3) and compared with the expert coding. The teacher talk move *press for reasoning* and the student talk move *asking for more information* performed the worst when compared with the expert coding. One probable reason for low performance could be very few instances belonging to these talk moves in the test set or a skewed distribution of the talk moves. Therefore, this needs to be explored further to determine the reason for low performance of these specific talk moves.

6. LIMITATIONS

The work described in this paper is novel and not without limitations. The first limitation is that we could not evaluate the classification model for Data Set 3 using Data Set 1, the model based on the full Accountable Talk framework from discourse in a traditional classroom. Our data was gathered from one-on-one online tutoring, which inherently prevents classification of talk moves that require additional community members as in the original dataset based on classroom discourse.

The second limitation to note is the limited amount of human-coded data to evaluate the classification of talk moves by the proposed model. We recognize that a larger percentage of human-coded validation data is desirable and plan to address this limitation in future work.

7. CONCLUSION

The academic discourse that occurs between teachers and students in the interest of knowledge creation has a rich history of research demonstrating the importance of each turn. Despite the importance of these classroom exchanges, providing feedback to teachers is nearly impossible due to the labor-intensive task of collecting and categorizing discourse according to an evidence-based framework such as the Accountable Talk framework [16].

Recent events have demonstrated that academic discourse is not restricted to traditional, whole group classroom instruction; therefore, it is equally important to evaluate the quality of discourse occurring in online, one-on-one tutoring sessions. The work described in this paper is both novel and promising as a means to reliably categorize teacher and student talk moves by applying machine learning models slightly modified from those validated on a complete framework. Future work is needed to further test and validate (with human coding) the machine learning model that was modified to accommodate one-on-one instruction (i.e., without talk moves for *accountability to the community*). Overall, this work has the potential to introduce a beneficial and simplified mechanism to provide feedback for teachers, thus affording strong potential to improve instructional practice and student learning outcomes.

8. ACKNOWLEDGMENTS

The research reported here was supported by the Institute of Education Sciences (IES), U.S. Department of Education through Grant R305C160004, the National Science Foundation (NSF) through Award# 2131052, and the Office of Naval Research (ONR) through Grant N000142012623. The opinions expressed are those of the authors and do not represent views of the IES, the NSF, or the ONR.

9. REFERENCES

- [1] Al-Adeimi, S., & O'Connor, C. (2021). Exploring the relationship between dialogic teacher talk and students' persuasive writing. *Learning and Instruction*, 71, 101388.
- [2] Bailey, D. H., Duncan, G. J., Murnane, R. J., & Au Yeung, N. (2021). Achievement gaps in the wake of COVID-19. *Educational Researcher*, 50(5), 266-275.
- [3] Candela, A. G., Boston, M. D., & Dixon, J. K. (2020). Discourse actions to promote student access. *Mathematics Teacher: Learning & Teaching*, 113, 266-277.
- [4] Cazden, C. B. (1988). *Classroom discourse: The language of teaching and learning* (2nd ed.). Portsmouth, NH: Heinemann.
- [5] Chin, C., & Osborne, J. (2010). Supporting argumentation through students' questions: Case studies in science classrooms. *Journal of the Learning Sciences*, 19, 230-284.
- [6] Correnti, R., Stein, M. K., Smith, M. S., Scherrer, J., McKeown, M., Greeno, J., & Ashley, K. (2015). Improving teaching at scale: Design for the scientific measurement and learning of discourse practice. In L. Resnick, C. Asterhan, & S. Clarke (Eds.), *Socializing intelligence through academic talk and dialogue* (pp. 303-320). Washington, DC: American Educational Research Association.
- [7] Howe, C., Hennessey, S., Mercer, N., Vrikki, M., & Wheatley, L. (2019). Teacher-student dialogue during classroom teaching: Does it really impact on student outcomes? *Journal of the Learning Sciences*, 28, 462-512.
- [8] Landis J. R., & Koch, G. G. (1977). The measurement of observer agreement for categorical data. *Biometrics*, 33(1), 159-174.
- [9] Lawrence, J. F., Crosson, A., Paré-Blagoev, E. J., & Snow, C. E. (2015). Word Generation randomized trial: Discussion mediates the impact of program treatment on academic word learning. *American Educational Research Journal*, 52, 750-786.
- [10] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... & Stoyanov, V. (2019). Roberta: A robustly optimized bert pre-training approach. arXiv preprint arXiv:1907.11692.
- [11] Mehan, H. (1979). *Learning lessons*. Cambridge, MA: Harvard University Press.
- [12] Mercer, N., & Dawes, L. (2014). The study of talk between teachers and students, from the 1970s until the 2010s. *Oxford Review of Education*, 40, 430-335.
- [13] Mercer, N., & Sams, C. (2006). Teaching children how to use language to solve maths problems. *Language and Education*, 20, 507-528.
- [14] Mercer, N., Wegerif, R., & Dawes, L. (1999). Children's talk and the development of reasoning in the classroom. *British Educational Research Journal*, 25, 95-111.
- [15] Michaels, S., O'Connor, C., & Resnick, L. B. (2008). Deliberative discourse idealized and realized: Accountable talk in the classroom and in civic life. *Studies in Philosophy and Education*, 27(4), 283-297.
- [16] Michaels, S., O'Connor, C., Hall, M. W., & Resnick, L. B. (2010). *Accountable talk® sourcebook*. Pittsburgh, PA: University of Pittsburgh Institute for Learning.
- [17] Michener, C. J., Proctor, C. P., & Silverman, R. D. (2018). Features of instructional talk predictive of reading comprehension. *Reading and Writing*, 31, 725-756.
- [18] Murphy, P. K., Wilkinson, I. A. G., Soter, A. O., Hennessey, M. N., & Alexander, J. F. (2009). Examining the effects of classroom discussion on students' comprehension of text: A meta-analysis. *Journal of Educational Psychology*, 101, 740-764.
- [19] National Council of Teachers of Mathematics (NCTM). (2014). *Principles to action: Ensuring mathematical success for all*. Reston, VA: National Council of Teachers of Mathematics.
- [20] National Governors Association Center for Best Practices (NGA Center) & Council of Chief State School Officers (CCSSO). (2010). *The common core state standards*. Washington, DC: NGA Center, CCSSO.
- [21] Puntambekar, S. (2022). Distributed scaffolding: Scaffolding students in classroom environments. *Educational Psychology Review*, 34, 451-472.
- [22] Suresh, A., Sumner, T., Huang, I., Jacobs, J., Foland, B., & Ward, W. (2018). Using deep learning to automatically detect talk moves in teachers' mathematics lessons. In 2018 *IEEE International Conference on Big Data* (pp. 5445-5447). IEEE.
- [23] Suresh, A., Sumner, T., Jacobs, J., Foland, B., & Ward, W. (2019, July). Automating analysis and feedback to improve mathematics teachers' classroom discourse. In *Proceedings of the AAAI conference on artificial intelligence* (Vol. 33, No. 01, pp. 9721-9728).
- [24] van de Pol, J., Volman, M., & Beishuizen, J. (2010). Scaffolding in teacher-student interaction: A decade of research. *Educational Psychology Review*, 22, 271-296.
- [25] Vygotsky, L. S. (1978). In M. Cole, V. John-Steiner, S. Scribner, & E. Souberman (Eds.), *Mind in society – The development of higher psychological processes*. Cambridge, MA: Harvard University Press.
- [26] Wissinger, D. R., & De La Paz, S. (2015). Effects of critical discussions on middle school students' written historical arguments. *Journal of Educational Psychology*, 108, 43-59.

Columns on Last Page Should Be Made as Close As Possible to Equal Length

Using Markov Models and Random Walks to Examine Strategy Use of More or Less Successful Comprehenders

Katerina Christhilf
Arizona State University, USA
k.christhilf@asu.edu

Natalie Newton
Arizona State University, USA
nnewton@asu.edu

Reese Butterfuss
Arizona State University, USA
rbutter2@asu.edu

Kathryn S. McCarthy
Georgia State University, USA
kmccarthy12@gsu.edu

Laura K. Allen
University of New Hampshire, USA
laura.allen@unh.edu

Joseph P. Magliano
Georgia State University, USA
jmagliano@gsu.edu

Danielle S. McNamara
Arizona State University, USA
danielle.mcnamara@asu.edu

ABSTRACT

Prompting students to generate constructed responses as they read provides a window into the processes and strategies that they use to make sense of complex text. In this study, Markov models examined the extent to which (1) patterns of strategies and (2) strategy combinations could be used to inform computational models of students' text comprehension. Random Walk models further revealed how consistency in strategy use over time was related to comprehension performance. High school ($n = 257$) and college students ($n = 153$) produced constructed responses at predetermined points while reading a scientific text. Each constructed response was scored for the presence of three common comprehension strategies (i.e., paraphrasing, bridging, elaborating), such that each constructed response could then be categorized as one of eight combination types. Markov chains revealed that more and less successful comprehenders leveraged different comprehension strategies, such that skilled comprehenders were more likely to use combinations of strategies while reading the text, particularly paraphrasing and making connections between ideas within the text (i.e., bridging). Random Walk analysis further demonstrated that successful comprehenders employed strategies more consistently. The results demonstrate the utility of Markov and Random Walk models in profiling learners' strategy use based on their constructed responses.

Keywords

Reading comprehension, Strategies, Markov models, Random Walk

1. INTRODUCTION

Reading is a fundamental life skill, whether one is trying to read a novel, a science textbook in a university course, a technical document for work, or an instruction manual for assembling furniture. However, many students struggle to comprehend texts. For K. Christhilf, N. Newton, R. Butterfuss, K. S. McCarthy, L. K. Allen, J. P. Magliano, and D. S. McNamara. Using Markov models and random walks to examine strategy use of more or less successful comprehenders. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 484–491, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852934>

example, 30% of U.S. students perform below basic proficiency in reading comprehension [24]. Several factors contribute to successful comprehension. To understand texts, particularly texts with complicated syntax or unfamiliar topics, readers engage in a variety of processes and strategies, including paraphrasing, bridging, and elaborating. When students paraphrase, they put the text in their own words. By contrast, bridging and elaborating require the generation of inferences that contain information that is not explicit in the text. Bridging refers to connecting ideas and information from different parts of the text. Bridging has been shown to be one of the most effective strategies for reading comprehension, particularly if the bridges connect more distant areas of text [18]. Elaborating is expanding on what one has learned from the text with one's prior knowledge about the topic.

Using bridging and elaborative inferences strategies has been found to improve comprehension on standardized reading tests [14]. More skilled readers are more likely to engage in more complex comprehension strategies, such as bridging and elaboration, that help them comprehend the text, while less skilled readers are more likely to rely on paraphrasing [7]. One way to encourage students to engage in such strategies is to ask them to self-explain a text as they read. Self-explanation prompts help students think deeper about the text, which facilitates strategy use and improves comprehension [25].

The strategies participants use while reading are typically examined via constructed response protocols (CRs). In CRs, students report their processes during a learning task. One method of obtaining CRs is asking participants to self-explain a text at specific points, and to type out these explanations. Prior work has shown that expert raters can reliably identify different comprehension strategies within CRs [17] and recent advances in natural language processing (NLP) have allowed researchers to develop and refine algorithms that can match these human judgments [4, 15]. Thus, mining readers' CRs can serve as a powerful tool to generate high quality learner models based on students' differing strategy use. The current study uses hand-coded CRs to examine participants' dynamic strategy use.

Several intelligent tutoring systems (ITSs) have been developed to enhance students' use of effective reading strategies. For example, iSTART [21] provides students with self-explanation reading training using a combination of instruction and practice via lesson videos, mini-games and reading practice. Training students on effective self-explanation reading strategies has been demonstrated

to improve students' text comprehension [20]. ITSs such as iSTART rely on the automated identification of different reading strategies in CRs to drive feedback to students on their self-explanations or other reading behaviors. The feedback may re-direct students to lessons on strategy use or provide recommendations of other strategies to leverage while reading. In order to better train students to modify their strategy use in ways that enhance reading comprehension, it is beneficial to understand the different ways students use strategies to engage with texts.

Past research more commonly examines paraphrasing, bridging, and elaboration strategies as independent variables or outcomes and measures strategy use in terms of their overall frequency of use [15, 23]. However, some types of strategy combinations may be more effective in promoting reading comprehension than using any one strategy in isolation. One possibility is that paraphrasing, by creating a more complete understanding of the most recently read text, facilitates the use of further strategies such as bridging and elaboration. The current study therefore examines combinations of strategy use within CRs, rather than overall frequencies of paraphrasing, bridging, and elaborating. Additionally, traditional analyses of CRs rely on an overall proportion or score across an entire text. This approach ignores the moment-to-moment changes in processing that occur as properties of texts change across sentences and paragraphs [22].

1.1 Our Approach

The current study aims to use two approaches that are more common in other areas to examine dynamic patterns in strategy use: Markov Chains and Random Walk analyses. These analyses are intended to reveal systematic differences in the ways that more and less successful comprehenders engage with texts. That is, the combinations and patterns of strategy use in students' CRs will vary in relation to their performance on a text comprehension assessment. More specifically, we predict that readers who more successfully comprehend a text will be marked by more frequent use of effective comprehension strategies, such as bridging and elaboration, and in particular, the use of combinations of strategies, such as para-bridging (i.e., combining paraphrasing and bridging within one CR). This prediction naturally stems from successful readers knowing how to use more strategies and use them in various contexts. Successful readers will also show a tendency towards certain types of switches, rather than randomly switching between strategies. Less successful comprehenders will be more likely to use strategies in isolation, especially paraphrasing, and will show less consistency in their strategy use patterns.

Markov chains are used to simulate processes with patterns that can be imprecisely predicted through probabilities [10]. In such processes, the probability of a given outcome depends on the previous outcome. For this study, the outcomes are different types of strategies, and the probability of using a strategy in a self-explanation depends on the strategy that was used in the previous self-explanation. The probabilities are determined based upon previously collected data and then displayed in a Markov chain visualization. Markov chains have been used in studies of mastery of ending letter sounds [12], and student knowledge and learning have been modeled using hidden Markov models in Bayesian Knowledge Tracing [6, 26, 29]. By contrast, Markov chains have not been widely explored as a method of studying comprehension processes in CRs. Cohesive features that help text comprehension, such as cues about causality, time, and space, are not always uniformly distributed within text; readers need to be sensitive to these features and adjust their strategy use accordingly to successfully comprehend a text [22]. Successful comprehenders are likely to be strategic in their

uses of comprehension processes, meaning they match their strategy use to the text [8]. Their strategy use is likely to be more structured, with consistent patterns in the types of switches that occur between strategies. Using Markov chains provides a learning analytic technique with strong potential to reveal students' flexibility in comprehension strategy use, as well as how those dynamic processes differ as a function of text comprehension performance.

To show the consistency of pattern use over time for individual participants, the current study uses random walk models, which are a type of sequential pattern analysis tool. They provide a spatial representation of an individual's path taken over time [2]. Such representations have commonly been used in the field of ecology, such as modeling animal migration patterns [3]. The movements of red deer are modeled on a graph with an x- and y-axis; every movement of the deer is plotted as a movement on the graph. Random walks have been used for a variety of other purposes, including decision making [9] and numerical cognition [5]. Using CRs allows us to code participants' strategy use and in turn, random walk models afford examining the consistency of students' strategy use over time.

Participants' random walks all start at (0, 0) on a graph. Four types of strategy use are defined as four different directions on the graph. Each CR counts as one step of the walk and moves one unit in a direction, based on the strategy used. The end point of each student's walk is plotted on a graph. Participants with more consistent strategy use have end points farther from the origin, while participants who frequently switch their strategy use have end points closer to the origin. Students who have more structured strategy use are likely to have some consistencies in the types of strategies they use, such as relying on paraphrasing and bridging. Inconsistent strategy use could indicate participants are not creating thoughtful explanations or using strategic choices for text comprehension. Their random walks may display multiple switches in strategy use and/or reliance on ineffective strategies. Random walk models offer additional information compared to Markov chains by showing the consistency of strategy use for individual participants, rather than the aggregated switching tendencies displayed by a Markov chain. Additionally, the random walk model is designed to show each participants' overall consistency over time, while Markov chains show probabilities of switching for each individual strategy.

The current study uses two extant datasets to help determine the generalizability of the research findings. Dataset 1 included 2322 CRs from 234 high school students (158 female; mean age = 17.58) [16]. Dataset 2 included 2448 CRs generated by 153 undergraduates. We predict that the results will be similar across datasets, but that the older, likely more skilled readers, in Dataset 2 would show more frequent use of effective strategies, more frequent use of combinations of strategies, and greater consistency of strategy use, compared to participants in Dataset 1.

2. METHOD

2.1 Datasets

In Dataset 1, participants read one of two texts, "Heart Disease" or "Red Blood Cells". Participants were asked to provide self-explanations at nine points in their assigned text. Both texts contain about 300 words and use scientific terminology suitable for the age of the participants. They have been validated in prior studies and matched for linguistic difficulty. In Dataset 2, participants read a text called "Cell Division", containing about 600 words, and were asked to self-explain the text at 16 different points. In both datasets, participants completed comprehension questions about the text after

reading. Performance on these questions serves as the dependent variable, which reflects students' comprehension success.

2.2 Expert Rating of Strategy Use

Verbal protocols were scored by expert raters using a self-explanation rubric that assesses the presence or absence of three reading strategies: paraphrase, bridging, and elaboration [17]. Paraphrase presence was scored based on the inclusion of idea units from the most recently read sentence. Bridging presence was scored based on the inclusion of references to other parts of the text. Elaboration presence was scored based on the inclusion of words or ideas that elaborate on the text but were not directly present in the text. Since paraphrase, bridging, and elaboration were scored independent of one another, the presence of any one strategy could be observed on its own or in combination with any of the other two strategies. Then, responses were categorized as containing one of eight strategy combination types, based on the presence of individual strategies: "None", "Paraphrase" (paraphrasing in isolation), "Bridge" (bridging in isolation), "Elaboration" (elaborating in isolation), "Para-bridge" (paraphrasing and bridging in combination), "Para-elab" (paraphrasing and elaborating in combination), "Bridge-elab" (bridging and elaborating in combination), and "All" (paraphrasing, bridging, and elaborating in combination). These categories did not include the order in which strategies were used within a single response, as the strategies were sometimes intermixed within a single response. For example, a "para-bridge" response means that paraphrasing and bridging were both used, but not in any particular order.

Scoring training was deemed complete when raters reached sufficient reliability for each coding category (weighted kappa ≥ 0.71). CRs were divided between pairs of raters such that each scored 60% of participant protocols, with 20% of participant protocols being scored by both raters. To calculate inter-rater reliability, protocols scored by both expert raters were compared per dataset, per text, and per coding category on a regular basis up to the latest point where both raters had scored the same protocols. Expert raters discussed coding disagreements to reach a consensus on the score. Following discussions, each rater independently reviewed and rescored as needed their entire set of protocols. Once raters completed scoring for a given text, scores were finalized only after raters achieved 0.71 or above weighted kappa per coding category.

2.3 Data Analysis

In both datasets, participants were divided into two groups based on the median performance (50%) in the text comprehension test. In Dataset 1, the low comprehension group included 102 participants, and the high comprehension group included 111 participants; 21 participants who scored on the median were excluded. In Dataset 2 (median = 50%), each group included 72 participants; 9 participants who scored on the median were excluded. The code used to create the Markov chains and the Random Walk displays can be found at <https://github.com/kchristilf/Markov-Chains>.

Markov chains were used to examine the dynamic nature of students' strategy use, as well as the overall frequency of different strategies. For each chain, we calculated the frequency of each type of strategy switch from one CR to the next. For instance, if a participant used paraphrasing in one CR and used para-bridging in the following submission, we added "1" to the overall frequency of switching from paraphrasing to para-bridging. The frequency of switches from a strategy were divided by the total number of switches from that strategy overall to calculate the probability of

switching from one strategy to the next. We also calculated the number of times each strategy was used overall.

These calculations were visibly represented in Markov chain diagrams created using 'Matplotlib' in Python (see Figures 2 and 3). Nodes were created to represent each strategy that was used more than five times. The area of the node was scaled relative to the overall number of times that strategy was used. The probability of switching from one strategy to another was represented by arrows between nodes. Wider arrows indicate a higher probability of switching from the origin node to the end node. Strategy use frequencies and transition probabilities are labeled when possible.

To examine consistency of strategy use over time, the current study uses random walk visualizations. As there are several possible "strategy moves" that a reader can make, we opted to simplify the analysis to four choices for visibility. Strategy use per trial was defined as using no strategies, paraphrasing in isolation, bridging in isolation, or two or more strategies in combination. The two or more strategies grouping includes para-bridging, para-elaborating, and para-bridge-elaborating. Each of these strategy combinations are an example of supplementing paraphrasing with one or more strategies, so we chose to combine them into a single group. For every participant's Random Walk analysis, each CR is coded as one "strategy move", and each strategy move is associated with a one unit movement in a specified direction. "No strategies" means one step down on the y-axis, "paraphrasing" means one step left on the x-axis, "bridging" means one step right on the x-axis, and "two or more strategies" means on step up on the y-axis. A single participant's full random walk is shown in Figure 1. The visualization shows that the participant primarily used paraphrasing or two or more strategies as their walk remains in the upper left quadrant. The student sometimes switched between different strategy types as shown by the relative distance away from the origin. In Figures 4 and 5, just the endpoints of each participant's walk were plotted onto a graph with an origin of (0, 0). For each random walk visualization, the participants' endpoint of the walk was graphed as a dot with 30% visibility, so that endpoints with more participants are darker. The graphs were created in Python using Matplotlib.

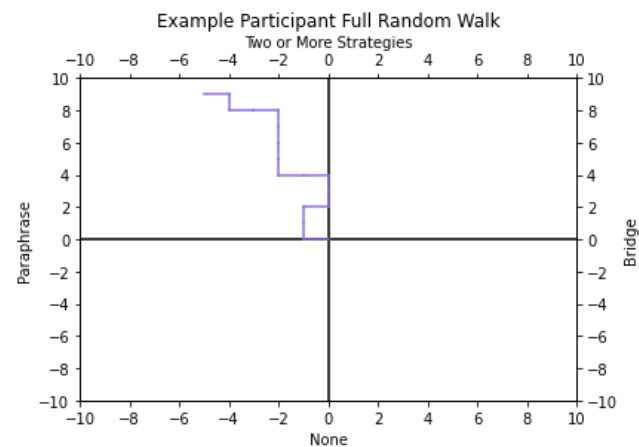


Figure 1. Example of a single participant's Random Walk.

3. RESULTS

3.1 Markov Chains

Elaboration in isolation and the combination of bridging and elaboration were not included as nodes in the Markov chains because participants used those strategies fewer than five times for each

chain. The six strategies that are represented in the Markov chains are no recorded strategies (labeled “None” or “N”), paraphrasing (“Para”), bridging (“Bridge” or “B”), paraphrasing and bridging in combination (“Para-Bridge”), paraphrasing and elaborating in combination (“Para-Elab” or “P-E”), and paraphrasing, bridging, and elaborating in combination (“All”).

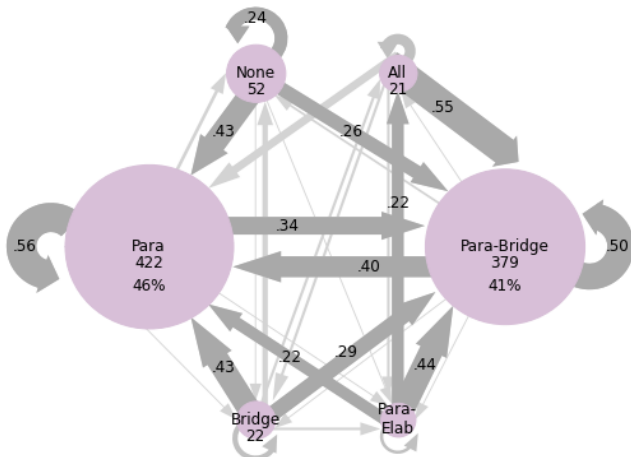


Figure 2a. Markov chain for low comprehension group in Dataset 1.

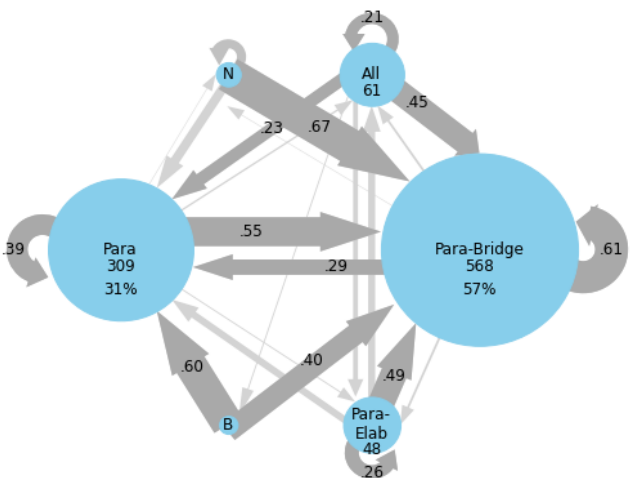


Figure 2b. Markov chain for high comprehension group in Dataset 1.

The variation in strategy use comparing more and less successful high school readers (Dataset 1) is shown in Figure 2. The low-comprehension group had high probabilities (>40%) for the following transitions: none to paraphrase (43%), paraphrase to paraphrase (56%), bridge to paraphrase (43%), para-bridge to para-bridge (50%), para-elab to para-bridge (44%), and all to para-bridge (55%). The high-comprehension group had high probabilities for none to para-bridge (67%), paraphrase to para-bridge (55%), bridge to paraphrase (60%), para-bridge to para-bridge (61%), para-elab to para-bridge (49%), and all to para-bridge (45%). The low comprehension group tended to paraphrase (n=503 instances) more than para-bridging (n=420). Paraphrasing was the only strategy students were more likely to continue using than to switch to another strategy from. Participants were also more likely to use none of the analyzed strategies (n=52) than all three (n=21). Meanwhile, the high comprehension group used more para-bridging (n=568) than paraphrasing (n=309), as shown in Figure 2. Para-bridging was the

only strategy students were more likely to continue using than to switch from. Students in the high comprehension group were more likely to use a combination of all three strategies (n=61) than no strategies (n=8). They were also more likely to switch to using para-bridging than the low comprehension group, even when their previous response used no strategies. Participants in the high comprehension group were more likely to use para-elaboration as a strategy than the low comprehension group (n=18, n=48). These results suggest paraphrasing is used by most readers. However, readers who are more successful at comprehending texts supplement paraphrasing with other strategies, particularly through para-bridging but also through para-elaborating and para-bridge-elaborating.

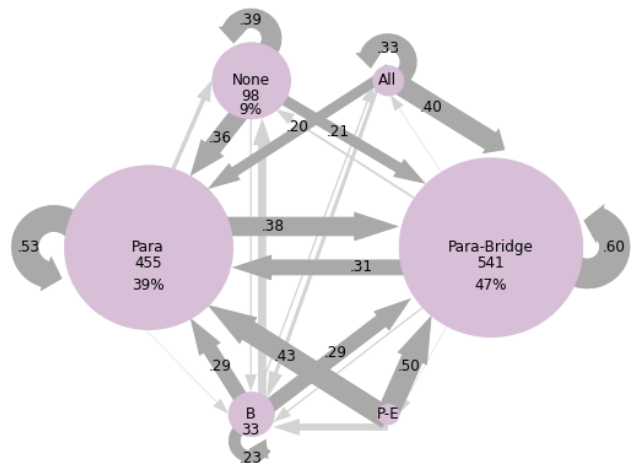


Figure 3a. Markov chain for low comprehension group in Dataset 2.

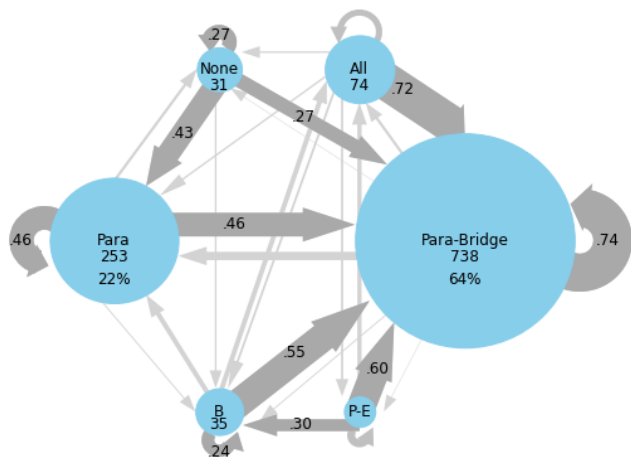


Figure 3b. Markov chain for high comprehension group in Dataset 2.

As predicted, the college students in Dataset 2 showed similar trends, as shown in Figure 3. The low-comprehension group had high probabilities (>40%) for the following transitions: paraphrase to paraphrase (53%), para-bridge to para-bridge (60%), para-elab to para-bridge (50%), para-elab to paraphrase (43%), and all to para-bridge (40%). Participants in the low comprehension group were more likely to use none of the analyzed strategies (n=98) than all three (n=15). Less successful comprehenders used paraphrasing in isolation more frequently than more successful comprehenders.

In the high comprehension group, more distinct trends emerge. Most of the high probability switch types involve para-bridging: the most common switches were none to paraphrase (43%), paraphrase to paraphrase (46%), paraphrase to para-bridge (46%), bridge to para-bridge (55%), para-bridge to para-bridge (74%), para-elab to para-bridge (60%), and all to para-bridge (72%). They were more likely to use a combination of all three strategies ($n=63$) than no strategies ($n=30$). Participants in the high comprehension group, compared to the low comprehension group, used more para-bridging ($n=738, n=541$) and were more likely to continue para-bridging from the previous CR (74% vs 60%). When paraphrasing in isolation, they were equally likely to switch to para-bridging as to continue paraphrasing in isolation (46%), compared to the low comprehension group that was more likely to continue paraphrasing in isolation (53%). More successful comprehenders seem to rely on the combination of paraphrasing and bridging. This suggests that the combination of paraphrasing and bridging is generally the most effective aid to comprehension, compared to other examined strategy combinations.

Overall, participants tended to not switch between using no strategies or paraphrasing and using all strategies. Participants were much more likely to switch from using a single strategy to two strategies or vice versa. Participants with high comprehension were more likely to switch between using one to three strategies, while participants with low comprehension were more likely to switch from using between zero to two strategies.

The two datasets have some differences from each other, likely due to the differences in age and education between the two groups. In Dataset 2, both groups were more likely to para-bridge than to paraphrase, while in Dataset 1, the low comprehension group was less likely to use para-bridging than paraphrasing. In both datasets, participants tended to continue to use the same strategy rather than switching. As para-bridging seems to be the more useful strategy in text comprehension, students in Dataset 2 may be more skilled, which can be expected given that they are university students, rather than high school students. The differences in strategy use between datasets may also be due to differences in the difficulty of the text used, as participants in Dataset 2 read a more complicated text, potentially requiring strategic processing to a greater extent.

One potential argument against examining participants' probabilities of switching between strategies is that switching is solely dependent on overall strategy use. For instance, one might argue that participants frequently switch to para-bridging because that strategy is used commonly in general. A series of chi-square analyses were conducted to determine if transitions were random (solely dependent on base frequencies of strategy use), or if there were meaningful trends in switching. Chi-squares were conducted for each switch type grouping (all switches from none, all switches from paraphrasing, etc.). Only transition types with an expected value of 5 or more were calculated (all $ps < .05$). In Dataset 1, for the low comprehension group 3 out of 6 chi-squares were significant; for the high comprehension group 2 out of 4 were significant. In Dataset 2, 3 out of 5 chi-squares in the low comprehension group and 4 out of 5 chi-squares in the high comprehension group were significant.

3.2 Random Walk

In Dataset 1 (Figure 4), each participant had 9 steps, leading to a maximum distance of 9 from the origin if the same strategy was repeatedly used. In Dataset 2 (Figure 5), each participant had 16 steps. Good comprehenders are shown in blue, while poorer comprehenders are shown in pink.

For Dataset 1, visual inspection reveals a broad pattern of readers ending in the upper right quadrant, replicating the findings from the Markov chains analyses such that readers tend to use paraphrase and para-bridge most commonly. Participants who comprehended the text less successfully showed less consistency in their strategy use than other participants, as shown in Figure 5 by the endpoints that are closer towards the origin. They were more likely to have switched to using bridging or no strategies. Those who were more consistently used paraphrasing or a combination of strategies. For Dataset 2, participants in the low comprehension group showed less consistency in their strategy use, compared to the high comprehension group. Low comprehension group participants were more likely to use no strategies and bridging compared to participants in the high comprehension group. Both groups showed a tendency towards using paraphrasing and using two or more strategies in combination. Dataset 2 has considerably more variability than Dataset 1. This may be due to the higher number of response opportunities compared to Dataset 1 (16 vs 9). Participants in Dataset 2 may have needed to use a greater variety of strategies to understand the more difficult text, or they may have experienced more fatigue towards the end of the task.

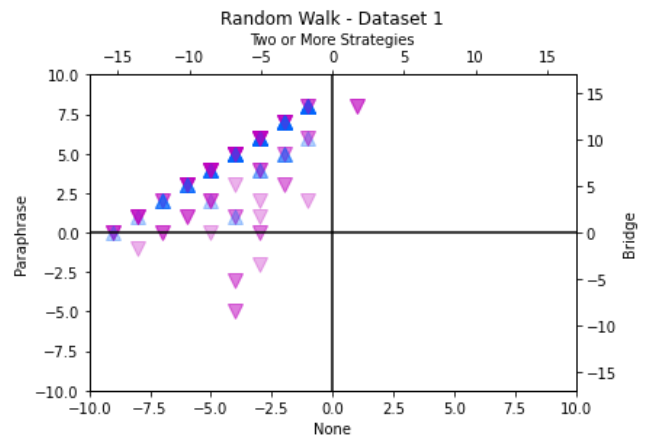


Figure 4. Random Walk for Dataset 1.

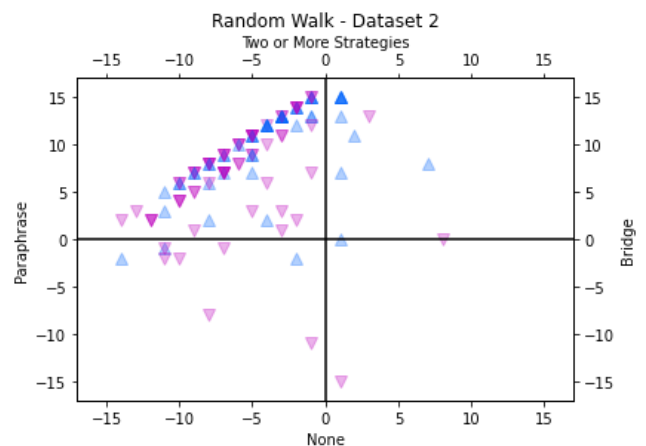


Figure 5. Random Walk for Dataset 2.

To quantify these Random walks, we calculated average Euclidean distance. Euclidean distance represents the distance of each step from the origin. It is used as a quantitative measure of the consistency of participants' strategy use. Pearson product-moment correlation tests showed that Euclidean distance was significantly

correlated with comprehension score (Dataset 1, $n = 234$, $r = .291$, $p < .001$; Dataset 2, $n = 153$, $r = .293$, $p < .001$). This suggests that readers who were more consistent in their strategy use (as reflected by greater distance from the origin) had better overall comprehension success.

4. DISCUSSION

Discourse comprehension researchers commonly examine overall frequencies of different types of strategies, rather than the dynamics of strategy use. The current study applied Markov chains, a methodology previously used in EDM, and Random Walk analysis, a methodology used in other domains, to better understand the dynamics of strategy use in reading comprehension. These data analytic techniques yielded qualitative evidence that more successful comprehenders engaged in markedly different strategy use patterns than their less-successful peers. For instance, Markov chains showed that high text comprehension performance was associated with use of multiple strategies, as well as switching to and from para-bridging. Poorer text comprehension was associated with using strategies in isolation, such as paraphrasing and bridging, and less structure in switching between strategies. Random walk analysis illustrated that students who are less successful in text comprehension are less consistent in their strategy use. They switch between using strategies in isolation, and at times use no strategies at all. On the other hand, more successful comprehenders rely on using paraphrasing and combinations of strategies that include paraphrasing, and they rarely switch to using solely bridging or no strategies. Understanding the dynamics of strategy use has been elusive [8], and so these types of analytic tools provide new insights about strategy use and how it differs across readers. These analyses show that solely examining the types of strategies used within constructed responses neglects the ways in which patterns of strategy use differ among readers.

Our results suggest that examining combinations of strategies may be useful for gaining a more complete picture of students' strategy use. Markov chains and Random Walk models show that readers who are comprehending texts successfully use multiple strategies while reading. They are frequently using multiple strategies within a single CR of a small section of text. Prior studies have shown that skilled readers are more likely to bridge than less-skilled readers. Examining combinations of strategies demonstrates that successful readers are not foregoing paraphrasing for better comprehension; they continue to ground their understanding of the text in paraphrasing, and frequently use other strategies to deepen comprehension. Because bridging is connecting information from different parts of the text, it contributes to a more coherent understanding of the text. However, it is difficult to connect different parts of text without a solid understanding of each of the individual parts. When the syntax, vocabulary, or concepts in a sentence are difficult, it may be challenging to build a mental representation of the sentence without paraphrasing it. This mental representation must be constructed prior to higher-order strategies such as bridging and elaborating. The results suggest that models of student strategy use should include the combinations of strategies students use, not just the individual types of strategies.

One limitation of the work is that some analyses relied on a median split. While such an approach is common for exploring differences between more and less successful comprehenders [11, 19], this dichotomy can mask within-group differences. There may also exist more than two types of patterns in strategy use. Future research should use a more nuanced approach to determining differences in strategy use and how they relate to reading comprehension. For example, we are currently exploring approaches such as k-means

clustering [26] to identify profiles of readers' use of combinations of strategies and transitions between strategies.

Similarly, the chi-square analyses used to assess the Markov chains are limited in their interpretability. Such tests show that there are differences in patterns, but not the full nature of those patterns. We are exploring approaches such as lag sequential analysis [8, 27], which could be conducted for the transitions that are embedded within the chi-square tests. However, this inherently comprises 10 to 16 transition types to test for significance, which potentially inflates Type I errors. Future work should thus leverage the current results to determine in advance the most common transition types to assess using lag-sequential analysis.

Finally, our analysis focuses on differences across comprehension of a specific text, rather than on relatively stable individual differences (e.g., reading skill). Future work should explore how readers' individual differences in comprehension-related factors such as reading skill [15], prior knowledge [28], and motivation [1] relate to strategy use and strategy patterns, and the extent to which strategy use on a given text influences text comprehension beyond general reading skill (as measured by an outside assessment, such as the Gates-MacGinitie Reading Test [13]). Further studies should also consider extending this work to other populations, such as second language learners and struggling readers.

The current study relies on human judgements of strategy use, which are formed through retroactive analysis of data. Researchers are in the process of refining NLP-driven algorithms that can detect these strategies during reading. The current work suggests the possibility of combining these detectors with Markov models and Random Walk analysis as a means of providing real-time stealth assessment of student learning and just-in-time support. These analyses can be used to create different profiles of learners and determine the types of strategy use that are most helpful to comprehension. These profiles could include the types of strategies used and the number of times users switch between different strategies. Once these profiles are established, real-time analysis of students' strategy use within their typed responses in an automated system can be leveraged to augment feedback. For example, if a student were demonstrating ineffective strategy patterns, an adaptive learning environment (or an instructor using a teacher interface) could intervene and encourage revising the self-explanation or scaffolding to help the student to use more effective strategies. Combining these data mining approaches with adaptive feedback would allow teachers to provide high-quality, real-time individualized instruction that can help a greater number and variety of students to become more successful comprehenders.

5. ACKNOWLEDGMENTS

We are grateful to Bryce Badaruddin, Kathryn Fleddermann, Emily Goblirsch, Elizabeth Kenney, Melanie Knezevic, Jacob Snyder, and Melissa Vasquez for their help in coding the constructed responses for strategy use. The research reported here was supported by the Institute of Education Sciences, U.S. Department of Education, through Grant R305A190063 to Arizona State University. The opinions expressed are those of the authors and do not represent views of the Institute or the U.S. Department of Education.

6. REFERENCES

- [1] Ahmadi, M. 2017. The Impact of Motivation on Reading Comprehension. *International Journal of Research in English Education*, 2 (Mar. 2017), 1-7. DOI=<http://dx.doi.org/10.18869/acadpub.ijree.2.1.1>

- [2] Benhamou, S., and Bovet, P. 1989. How animals use their environment: a new look at kinesic. *Animal Behavior*, 38, 3 (Sep. 1989), 375-383. DOI= [https://doi.org/10.1016/S0003-3472\(89\)80030-2](https://doi.org/10.1016/S0003-3472(89)80030-2)
- [3] Berthelot, G., Saïd, S., & Bansaye, V. 2021. A random walk model that accounts for space occupation and movements of a large herbivore. *Scientific Reports*, 11 (Jul. 2021), 14061. DOI= <https://doi.org/10.1038/s41598-021-93387-2>
- [4] Cai, Z., Graesser, A., Forsyth, C., Burkett, C., Millis, K., Wallace, P. Halpern, D., & Butler, H. 2011. Dialog in AR-IES: User Input Assessment in an Intelligent Tutoring System. In *Proceedings of the 3rd IEEE International Conference on Intelligent Computing and Intelligent Systems*. IEEE Press, 429-433. DOI= <http://dx.doi.org/10.13140/2.1.4284.5446>
- [5] Cohen, D. J. & Quinlan, P. T. 2016. How numbers mean: Comparing random walk models of numerical cognition varying both encoding processes and underlying quantity representations. *Cognitive Psychology*, 91 (Dec. 2016), 63-81. DOI= <https://doi.org/10.1016/j.cogpsych.2016.10.002>
- [6] Corbett, A. T. & Anderson, J. R. 1995. Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. *User Modeling and User-Adapted Interaction*, 4 (Dec. 1994), 253-278. DOI= <https://doi.org/10.1007/BF01099821>
- [7] Coté, N., Goldman, S. R., & Saul, E. U. 1998. Students making sense of informational text: Relations between processing and representation. *Discourse Processes*, 25, 1, 1-53. DOI= <https://doi.org/10.1080/01638539809545019>
- [8] Cromley, J. G. & Wills, T. W. 2016. Flexible strategy use by students who learn much versus little from text: transitions within think-aloud protocols. *Journal of Research in Reading*, 39, 1 (Feb. 2016), 50-71. DOI= <https://doi.org/10.1111/1467-9817.12026>
- [9] Farrell, S. & Lewandowsky, S. 2010. Computational Models as Aids to Better Reasoning in Psychology. *Current Directions in Psychological Science*, 19, 5 (Oct. 2010), 329-335. DOI= <https://doi.org/10.1177/0963721410386677>
- [10] Geyer, C. J. 1992. Practical Markov Chain Monte Carlo. *Statistical Science*, 7, 4 (Nov. 1992), 473-483. DOI= <https://www.jstor.org/stable/2246094>
- [11] Goldman, S. R., Braasch, J. L. G., Wiley, J., Graesser, A. C., & Brodowinska, K. 2012. Comprehending and learning from internet sources: Processing patterns of better and poorer learners. *Reading Research Quarterly*, 47, 4, 356-381.
- [12] Kaplan, D. 2008. An overview of Markov chain methods for the study of stage-sequential developmental processes. *Developmental Psychology*, 44, 2 (Mar. 2008), 457-467. DOI= <https://doi.org/10.1037/0012-1649.44.2.457>
- [13] MacGinitie, W. H., & MacGinitie, R. K. 1989. Gates-MacGinitie Reading Test (3rd ed.). Itasca, IL: Riverside.
- [14] Magliano, J. P., Higgs, K., Santuzzi, A., Tonks, S. M., O'Reilly, T., Sabatini, J., Feller, D., Kopatich, R. D., Ray, M., & Parker, C. 2020. Testing the inference-mediation hypothesis in a post-secondary context. *Contemporary Educational Psychology*, 61 (Apr. 2020), 101867. DOI= <https://doi.org/10.1016/j.cedpsych.2020.101867>
- [15] Magliano, J. P., Millis, K. K., The RSAT Development Team, Levinstein, I., Boonthum, C. 2010. Assessing comprehension during reading with the Reading Strategy Assessment Tool (RSAT). *Metacognition and Learning*, 6 (Dec. 2010), 131-154. DOI= <https://doi.org/10.1007/s11409-010-9064-2>
- [16] McCarthy, K. S., Likens, A. D., Johnson, A.M., Guerrero, T. A., & McNamara, D. S. 2018. Metacognitive Overload!: Positive and Negative Effects of Metacognitive Prompts in an Intelligent Tutoring System. *International Journal of Artificial Intelligence in Education*, 28 (Feb. 2018), 420-438. DOI= <https://doi.org/10.1007/s40593-018-0164-5>
- [17] McCarthy, K. S., Magliano, J. P., Snyder, J. O., Kenney, E. A., Newton, N. N., Perret, C. A., Knezevic, M., Allen, L. K., & McNamara, D. S. 2021. Quantified qualitative analysis: Rubric development and inter-rater reliability as iterative design. In *15th International Conference of the Learning Sciences* (Bochum, Germany, June 2021). ICLS '21, 139-146. International Society of the Learning Sciences. URL= <https://repository.isls.org/handle/1/7458>
- [18] McCrudden, M. T., Huynh, L., Lyu, B., & Kulikowich, J. M. 2021. Bridging Inferences and Learning from Multiple Complementary Texts. *Discourse Processes*, 58, 5-6 (May 2021), 529-548. DOI= <https://doi.org/10.1080/0163853X.2021.1924586>
- [19] McMaster, K. L., van den Broek, P., Espin, C. A., White, M. J., Rapp, D. N., Kendeou, P., Bohn-Gettler, C. M., Carlson, S. Making the right connections: Differential effects of reading intervention for subgroups of comprehenders. *Learning and Individual Differences*, 22, 1 (Feb. 2012), 100-111. <https://doi.org/10.1016/j.lindif.2011.11.017>
- [20] McNamara, D. S. 2017. Self-Explanation and Reading Strategy Training (SERT) Improves Low-Knowledge Students' Science Course Performance. *Discourse Processes*, 54, 7, 479-492. <https://doi.org/10.1080/0163853X.2015.1101328>
- [21] McNamara, D. S., Levinstein, I., & Boonthum, C. 2004. iSTART: Interactive Strategy Training for active reading and thinking. *Behavior Research Methods, Instruments, & Computers*, 36 (May 2004), 223-233. DOI= <https://doi.org/10.3758/BF03195567>
- [22] McNamara, D. S., & Magliano, J. 2009. Toward a comprehensive model of comprehension. *Psychology of Learning and Motivation*, 51, 297-384. DOI= [https://doi.org/10.1016/S0079-7421\(09\)51009-2](https://doi.org/10.1016/S0079-7421(09)51009-2)
- [23] Munoz, B., Magliano, J. P., Sheridan, R., & McNamara, D. S. 2006. Typing versus think-aloud when reading: Implications for computer-based assessment and training tools. *Behavior Research Methods*, 38, 2 (May 2006), 211-217. DOI= <https://doi.org/10.3758/BF03192771>
- [24] National Assessment of Educational Progress. 2019. *NAEP report card: 2019 NAEP reading assessment*. U.S. Department of Education, Institute of Education Sciences. URL= <https://www.nationsreportcard.gov/highlights/reading/2019/>
- [25] Neubrand, C. & Harms, U. 2017. Tackling the difficulties in learning evolution: effects of adaptive self-explanation prompts. *Journal of Biological Education*, 51, 4, 336-348. DOI= <https://doi.org/10.1080/00219266.2016.1233129>
- [26] Nguyen, H., Hou, X., Stamper, J. & McLaren, B. 2020. Moving beyond Test Scores: Analyzing the Effectiveness of a Digital Learning Game through Learning Analytics. In *Proceedings of the 13th International Conference on Educational Data Mining* (Online, Jul. 10-13, 2020). EDM '20. Anna N. Rafferty, Jacob Whitehill, Cristobal Romero,

Violetta Cavalli-Sforza (eds), 487-495. URL= <https://educationaldatamining.org/edm2020>

- [27] Pohl, M., Wallner, G., Kriglstein, S. 2016. Using lag-sequential analysis for understanding interaction sequences in visualizations. *International Journal of Human-Computer Studies*, 96 (Dec. 2016), 54-66. DOI= <https://doi.org/10.1016/j.ijhcs.2016.07.006>
- [28] Williams, J. J. & Lombrozo, T. 2010. Explanation constrains learning, and prior knowledge constrains explanation. In

Proceedings of the 32nd Annual Conference of the Cognitive Science Society. Cognitive Science Society.

- [29] Williamson, K., and Kizilcec, R. 2021. Effects of Algorithmic Transparency in Bayesian Knowledge Tracing on Trust and Perceived Accuracy. In *Proceedings of the 13th International Conference on Educational Data Mining* (Paris, France, Jul. 10-13, 2020). EDM '20. Anna N. Rafferty, Jacob Whitehill, Cristobal Romero, Violetta Cavalli-Sforza (eds), 338-344. URL= <https://educationaldatamining.org/edm2021>

No Meaning Left Unlearned: Predicting Learners' Knowledge of Atypical Meanings of Words from Vocabulary Tests for Their Typical Meanings

Yo Ehara
Tokyo Gakugei University
ehara@u-gakugei.ac.jp

ABSTRACT

Language learners are underserved if there are unlearned meanings of a word that they think they have already learned. For example, “circle” as a noun is well known, whereas its use as a verb is not. For artificial-intelligence-based support systems for learning vocabulary, assessing each learner’s knowledge of such atypical but common meanings of words is desirable. However, most vocabulary tests only test the typical meanings of words, and the texts used in the test questions are too short to apply readability formulae. We tackle this problem by proposing a novel dataset and a flexible model. First, we constructed a reliable vocabulary test in which learners answered questions regarding typical and atypical meanings of words. Second, we proposed a simple but powerful method for applying flexible and context-aware masked language models (MLMs) to learners’ answers in the above-mentioned vocabulary test results. This is a personalized prediction task, in which the results vary among learners for the same test question. By introducing special tokens that represent each learner, our method can reduce the personalized prediction task to a simple sequence classification task in which MLMs are applicable. In the evaluation, item response theory (IRT)-based methods, which cannot leverage the semantics of test questions, were used as baselines. The experimental results show that our method consistently and significantly outperformed the IRT-based baselines. Moreover, our method is highly interpretable because one can obtain the learners’ language abilities from the first principal component scores of the token embeddings representing each learner.

Keywords

Second Language Learning, Item Response Theory, Masked Language Models, Natural Language Processing

1. INTRODUCTION

In intelligent tutoring systems, it is important to accurately identify what is known and what is unknown to the learner

Y. Ehara. No meaning left unlearned: Predicting learners’ knowledge of atypical meanings of words from vocabulary tests for their typical meanings. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 492–499, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853169>

to recommend learning items according to the learner’s characteristics, such as the learner’s ability. When there are many learning items, it is difficult to test them all. In this case, only some of the knowledge is tested, and the test results are used to predict the other held knowledge.

This is especially fitting to vocabulary learning support systems for foreign languages. Because of the large vocabulary of a foreign language, research has been conducted in the fields of natural language processing (NLP) and applied linguistics to test only some words and to predict the held knowledge of other words based on the test results [19, 15, 10, 9, 8, 16]. As a result, standardized test questions have been developed to test learners’ knowledge.

However, for polysemous words, where a word has multiple meanings, the task of testing knowledge of only typical meanings and predicting the held knowledge of unexpected meanings from only the test results is a challenge. Because polysemous words are especially common in **high-frequency words** (words with high frequency in large corpora) that foreign language learners learn early on, it is also important for learners to properly grasp and understand the meaning of these words in context. However, many studies on foreign language learning support have focused on vocabulary size, and this area has been relatively unexplored. To the best of our knowledge, standardized questions that test the meanings of polysemous words have not been developed.

In the case of polysemous words, it is difficult to test for meanings other than the most typical ones. This can be attributed to the large number of words that must be learned; testing a learner’s knowledge of a single word more than once would impose a greater testing burden on learners. Hence, learners may often fail to learn word meanings other than the most representative meaning.

To improve learners’ language abilities, it is important to ensure that they learn all major meanings of polysemous words. In the development of artificial intelligence (AI)-based systems that support language learners by identifying and recommending such unlearned meanings, it is essential to identify the meanings of polysemous words that a given learner already knows. However, considering the testing burden, sufficient time is available to test only a few additional word meanings. Given that vocabulary tests use numerous words to measure a learner’s overall vocabulary, most typical vocabulary tests query only the major meanings of polyse-

It was a difficult period.
a) question
b) time
c) thing to do
d) book

(a) An example question in the vocabulary size test [3], which tests typical meanings of words.

She had a missed _____.
a) time
b) period
c) hour
d) duration

(b) An example of a question that tests atypical meanings of a word.

Figure 1: Outline of the task. Our goal is to predict whether a learner knows this word (i.e., can correctly answer the question) from the results of typical meanings of words to prevent these meanings from being left unlearned.

mous words. Therefore, it is desirable to be able to predict the extent to which a learner knows the non-representative meanings of a word by using existing test results for major word meanings. For example, it should be possible to predict whether a learner who knows the meaning of “figure” referring to a number also knows the meaning of “figure” referring to a person.

Intuitively, one can easily think of a two-step approach in which one first predicts the meaning of a polysemous word in a running sentence and then assigns the difficulty of each meaning of a word. However, this approach is impractical because the categorization of meanings by linguists is not typically designed for language learning. For example, Figure 1 show examples of the word “period” used in distinctly different contexts. However, linguistic categorization can be too fine-grained for language learners. For example, WordNet [12], one of the most carefully designed thesauri for the English language, separately lists “period” as a geological period as being a different meaning from “period” as a timespan. This is counter-intuitive for language learners and impractical for estimating the difficulty of “period” as a geological period separately from that as a timespan. Hence, another approach is necessary in which one directly predicts how likely it is that the language learner understands the meaning of a word used in an input sentence.

To the best of our knowledge, no existing datasets or methods have been provided in the literature to evaluate the extent to which such problems can be solved. Because it is presumably difficult to capture the meanings of words in running texts, these types of questions have not been extensively studied in vocabulary testing studies in applied linguistics [19, 15, 20]. Therefore, in this study, we propose a dataset and methods to evaluate how well these problems can be solved. Figure 1 are examples of the dataset. To this end, we used deep transfer learning state-of-the-art neural language models (MLMs), namely masked language models such as bidirectional encoder representations from transformers (BERT) [5]. Deep-learning-based NLP techniques cannot simply be applied to this personalized prediction problem in which different predictions must be made for each learner, even for the same given input sentence. Although recent educational AI studies also used BERT [24, 23, 25], they did not address this issue because they do not deal with personalized prediction tests.

In the proposed method, we demonstrate a simple approach to reduce this personalized prediction problem to a typical sequence classification problem in NLP by adding spe-

cial tokens that represent learners in language models. In our experiments, the prediction performance of the proposed method was superior to that of other methods such as item response theory (IRT) by a statistically significant margin.

Our method also showed high interpretability matching that of the IRT models: one merit of using IRT models is that they are highly interpretable, e.g., capable of extracting the ability estimates from the model, which BERT models cannot do. We showed that the first principal component scores of the embedding of the tokens representing each learner had a statistically significant correlation with the learners’ ability values obtained using the IRT.

The contributions of this paper are as follows.

- We focused on the importance of predicting whether language learners know the atypical meanings of a word and developed an evaluation dataset for this purpose.
- In addition, we proposed a simple method for applying deep transfer learning techniques to the aforementioned personalized prediction problem by introducing tokens that represent learners. The prediction performance of the proposed method is superior to that of IRT by a statistically significant margin.
- Finally, we demonstrated that with the proposed method, we can easily obtain learners’ ability values by using the first principal component scores of token embedding. This indicates that our method is highly interpretable, and hence suitable for educational use.

2. RELATED WORK

In educational data mining, [4] models the *spaced repetition*, or students’ memorization of learning items repeatedly, by extending statistical models. While [4] focuses on the temporal aspect in the process of learning second language vocabulary, we focus on predicting each learner’s knowledge of atypical meanings of words from the test results of typical meanings of words, considering text semantics and contexts. In AI in education, [1] addressed a case study of the personalized English vocabulary learning of 37 Syrian refugees using a language learning application called SCROLL. This study did not address the methodology or algorithm used in SCROLL but the case study of using it. While [7] applied BERT to readability prediction, [7] was not personalized.

Vocabulary test datasets were previously published via self-report testing [11, 18], or multiple-choice testing [6]. However, to the best of our knowledge, no reliable vocabulary

test datasets in which typical and atypical meanings of words have both been tested have been published.

Other datasets seemingly similar to our dataset include the SLAM dataset [22], which is based on responses to questions on the language learning application Duolingo, and the Complex Word Identification (CWI) dataset [26], where a large number of language learners were asked to annotate unfamiliar words in a sentence. The difference between these datasets and ours is that each subject answered only a small portion of the many questions.

3. VOCABULARY TESTING DATASETS

This section elaborates on our dataset. [6] is a publicly available dataset of vocabulary test results of language learners. However, it does not include questions regarding any typical/atypical meanings of words. Nevertheless, for comparability, we adopted their settings to build our dataset. Our dataset was compiled from the crowdsourcing service Lancers¹. To find learners who had some interest in learning English, only learners who had taken the Test of English for International Communication (TOEIC) test² in the past were permitted to take the vocabulary test. As a result, 235 subjects responded to the questionnaire. Because most learners in Lancers are native Japanese speakers, the native language of the learners was also assumed to be Japanese.

For typical vocabulary test questions, we used vocabulary size test (VST) [3], as [6] did. However, unlike [6], our focus is highly frequent words. In VST, the questions are ordered by the frequency of the corpus that [3] used for making VST. To reduce the test burden on the participants and to easily collect accurate answers, we eliminated 30 low-frequency words from the test. The remaining 70 questions were used for a typical vocabulary test. An example of these questions is shown in Figure 1 (a). The word(s) being tested are underlined in the sentence. The subject is asked to choose the option that is closest in meaning to the original sentence when the word(s) being tested are replaced. All options were designed to be grammatical when replaced.

We developed 13 questions that tested atypical meanings of words as follows. First, a computer science researcher, who was a non-native but fluent English speaker, drafted the questions. Second, English professors, namely two native English speakers (i.e., an American English speaker and an Australian English speaker) and a non-native speaker, checked and edited the questions for validity. In the actual examples of these two test sets in Figure 1 (b), the word “period” has a physiological meaning in addition to the usual meaning as a timespan. The subjects were asked to answer 13 questions, such as Figure 1 (b) before the 70 lexical test of typical usage. We included one question with an unexpected meaning but without a corresponding question on the group of typical word meaning questions. Therefore, the number of question pairs was 12. More details of our dataset are provided in the Appendix.

4. ITEM RESPONSE THEORY

¹<https://lancers.co.jp/>

²<https://www.ets.org/toEIC>

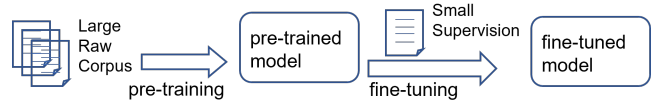


Figure 2: Deep transfer learning procedure

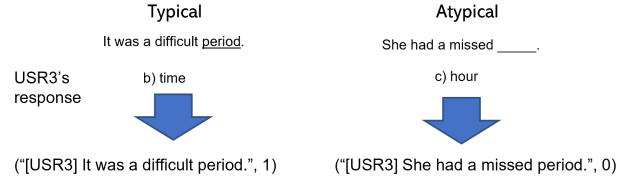


Figure 3: Proposed method to enable personalized prediction using BERT

This section briefly describes the IRT models. Let the number of subjects be J and the number of questions (or items) be I . For simplicity, we identify the index of the subject with the subject and the index of the item with the item. For example, the I -th item is simply written as I . We assume that y_{ij} is 1 when subject j answers item i correctly, and 0 when the subject answers incorrectly. Given the test result data $\{y_{ij} | i \in \{1, \dots, I\}, j \in \{1, \dots, J\}\}$, the 2PL model models the probability that subject j answers item i correctly with the following equation

$$P(y_{ij} = 1 | i, j) = \sigma(a_i(\theta_j - d_i)) \quad (1)$$

Here, σ is the logistic sigmoid function defined as $\sigma(x) = \frac{1}{1 + \exp(-x)}$. where the σ is a monotonically increasing function with $(0, 1)$ as its value range and $\sigma(0) = 0.5$. The $\sigma(x)$ is used to project real numbers into the range of $(0, 1)$ and treat them as probabilities. In (1), θ_j is called the *ability parameter*, and is a parameter that represents the ability of the subject. d_i is the difficulty parameter representing the difficulty of the item. From (1), when θ_j is greater than d_i , the probability of the subject answering correctly is higher than that of answering incorrectly. The value $a_i > 0$ is usually positive and is called the *discrimination parameter*. The larger this value, the more $\theta_j - d_i$ affects the probability of correct or incorrect answers. It is called “discrimination” because $\theta_j - d_i$ makes it easier to distinguish whether subject j will answer question i correctly or not. More intuitively, this indicates that question i is a good question in that it can accurately distinguish between learners with high ability and those with low ability.

5. PROPOSED METHOD

5.1 Deep Transfer Learning

In this section, we describe our proposed method. The proposed method is based on Transformer models such as BERT [5]. Transformer models employ transfer learning to capture the semantics of texts written in natural language Figure 2. First, a **pre-trained model** is prepared using large raw (i.e., unannotated) texts. This model can be trained using raw texts written by native English speakers, such as the Wikipedia text. This procedure is called **pre-training**. Typically, pre-training incurs high computational costs. Hence, we downloaded and prepared publicly available pre-trained

models. Each model is identified by an ID such as **bert-large-cased**, which denotes that the model was trained on a large Wikipedia corpus in a case-sensitive manner. Importantly, a pre-trained model can be used for many tests depending on the **fine-tuning**.

Then, an additional **fine-tuning** is performed to train the pre-trained model to the intended task. To this end, a small annotated corpus must be prepared for use in a supervised learning procedure. Such corpora are generally costly to construct; in this case, the corpus comprised the results of a vocabulary test. After the model is fine-tuned, it can be used to make predictions based on new input sentences.

5.2 Reducing the Personalized Prediction into Sequence Classification

Given a language learner taking a test and a word in a sentence, as shown in Figure 1 (b), our goal is to predict whether the test-taker knows the word. Notably, this is a *personalized* prediction; the prediction results differ among individual learners. In contrast, deep transfer learning does not support personalized predictions. Publicly available pre-trained models are preferable because pre-training is costly. Moreover, designing a new model that can achieve high performance using the available pre-trained models is relatively difficult. Thus, reducing the personalized prediction problem to a typical NLP task can be a practical solution, instead of developing a novel neural model for this task.

We reduced the personalized prediction task into a sequence classification task as shown in Figure 3. Sequence classification is a task in which a classifier takes a sequence or text as the input and predicts its label. Hence, to train the classifier, pairs of text and associated labels are used to constitute a small corpus for supervised learning in the fine-tuning phase. Thus, to use sequence classifiers in this task, we first need to convert the original vocabulary test result dataset into a sequence classification dataset so that sequence classifiers can handle the dataset.

In the example of Figure 3, **USR3**, the test-taker whose ID was 3 answered correctly on a multiple-choice question of the typical meaning of the word “period,” but incorrectly answered a question on the atypical meaning. To convert this record into a format that accepts a sequence classifier, we added special tokens: **[USRn]**.

Here, **[USRn]**, where n is replaced by the test-taker ID, represents each test-taker, or learner (user). By placing this at the beginning of the sequence, we notify the classifier that we want to predict the response of the test-taker specified by this token. Therefore, the example in Figure 3 shows that we aim to predict the response of **USR3** to the sequence “It is a difficult period.” In this example, as **USR3** answered the multiple-choice question correctly, the label for the question was set to 1. The rationale behind this conversion is that the test-taker could read the sentence if the test-taker answered the question correctly. Hence, the label denotes that the test-taker was able to successfully read the short sentence “It is a difficult period.”

Likewise, **USR3**’s answer to an atypical question can be converted to the sequence on the right-hand side of Figure 3. In

this example, as **USR3** answered incorrectly, the label for the question was set to 0. The option that is incorrectly chosen by **USR3** (“hour” in the example of Figure 3) is ignored in the sequence. This ensures a fair and accurate comparison, because IRT-based methods also do not consider the *incorrect* options or distractors chosen by the test-taker. Rather, they only consider whether a test-taker chooses the correct option. As each of these tokens represents a test-taker, there are as many tokens as the number of test-takers, starting from **[USR1]**.

Thus, the dataset can be converted into a sequence classification format. In the **transformers** library, the tokens used for Transformer models can be added using the **add_tokens** function. After conversion, we simply use the **AutoModelForSequenceClassification** to construct sequence classifiers. Note that this conversion enables BERT to handle personalized prediction by introducing **[USRn]** tokens. We introduce special tokens for each user and insert the token into the beginning of the sentence.

6. IRT-BASED ANALYSIS

To obtain the difficulty and discrimination parameters for IRT, we used the **pyirt** Python library³. This library was developed to conduct IRT analyses using marginalized maximum likelihood estimation (MMLE) [2, 21]. For the dataset described above, we used the 2PL model to obtain the above-mentioned parameters. The dataset includes 12 pairs of questions, such as Figure 1. The difficulty parameters for the usual and unexpected examples are shown on the horizontal and vertical axes, respectively, and plotted at the same scale and range on the horizontal and vertical axes in Figure 4. Each point represents a single word.

A dotted diagonal line is shown from the lower left to the upper right of Figure 4. The horizontal and vertical axes of Figure 4 represent the values of the difficulty parameter; the higher the value, the more difficult the task was judged to be. The point to the upper left of the diagonal line indicates that the difficulty level of an example that seemed unexpected to the learner was higher than that of the typical example. Moreover, the word was judged to be more difficult for the learner to correctly answer questions from the vocabulary test data. The results of the Wilcoxon test showed that the column of values on the vertical axis was larger than that on the horizontal axis by a statistically significant margin ($p < 0.01$), suggesting that the vertical-axis questions were more difficult than the horizontal-axis questions.

Discrimination was also analyzed. The plot is omitted for space limitation. Atypical meanings are expected to be less discriminating than typical meanings, because even high-ability learners may not know the correct answers of atypical meanings, whereas low-ability learners may know them. This tendency was observed as follows: For all words, it was estimated that the discrimination of typical examples was higher than that of unexpected examples. This result was found to be statistically significant using the Wilcoxon test ($p < 0.01$).

7. EXPERIMENTS OF PREDICTIONS

³<https://github.com/17zuoye/pyirt>

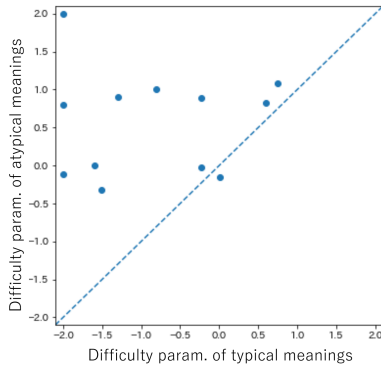


Figure 4: Plot of the difficulty of a typical meaning (horizontal) vs. that of an atypical meaning (vertical) for each word.

7.1 Item Response Theory-Based Settings

A naive method to deal with the difficulty of an atypical meaning of a word is to simply regard its difficulty as being the same as that of its typical meaning. Hence, we measured the negative effects of substituting the difficulty of the atypical meanings of a word with the difficulty of its typical meanings on predicting the subjects’ responses.

To investigate this, we conducted the following experiment, as shown in Figure 5. First, we divided 235 subjects into 135 and 100 subjects. We estimated the parameters without using the responses of the latter 100 subjects for the 12 questions (1,200 responses). The parameters of the 12 atypical example questions were estimated only from the responses of the former 135 subjects, while the parameters of the 70 typical example questions were estimated from the responses of all 235 subjects. From (1), we can see that the estimated values of subject ability θ_j and the difficulty of the example d_i are sufficient to predict if subject j answers question i correctly or not by checking if $\theta_j > d_i$ or not, respectively. Hence, once all θ_j and d_i can be estimated, we can make predictions. For the 12 typical and atypical question pairs, we have two prediction methods: one that uses the difficulty parameters of the typical examples of the pairs for d_i and one that uses that of atypical examples of the pairs. Thus, we compared the prediction accuracy of these 1,200 responses.

Several methods are conventionally used to estimate the parameters of IRT models. As in the previous sections, we used the pyirt library, which implements MMLE, for parameter estimation. MMLE assumes that the ability parameter can only take several values to marginalize. This causes stepwise shapes in the resulting ability parameter plots.

7.2 Our Settings

We constructed a BERT-based personalized predictor [5]. For the neural classification, we used the same settings described in Section 5.2. As described in Table 1, we compared the pre-trained Transformer models, all of which were publicly available from the HuggingFace website⁴. Then, we conducted a *fine-tuning* by using our own data.

⁴<https://huggingface.co/>

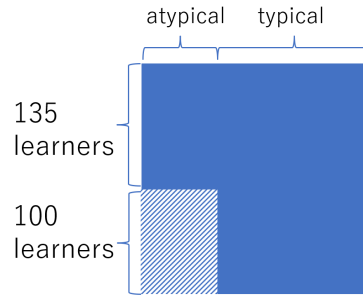


Figure 5: Experiment setting. Filled areas are *training* data, i.e., used for estimating the parameters. All methods are evaluated based on the accuracy of the responses of the dashed area, i.e., the responses for atypical words of the 100 *test* learners.

Table 1: Transformer Models Used for Experiments

Model Name	Model	cased/uncased
bert-base-cased	BERT [5]	cased
bert-base-uncased	BERT [5]	uncased
bert-large-cased	BERT [5]	cased
roberta-base	RoBERTa [17]	cased
albert-base-v2	ALBERT [14]	uncased

Here, we describe how we converted the personalized prediction of whether a learner knew a given word into the task of sequence classification and its fine-tuning. Sequence classification is a supervised classification task in which the goal is to predict labels by using a sequence as an input. Here, the label is 1 if the learner knows the meaning of the word, i.e., answered the question about the meaning of the word correctly; otherwise, the label is 0.

Because the aim is to make personalized predictions, it is necessary to incorporate learner test-takers in the sequence. Thus, we added special tokens to represent individual learners. For example, if the sequence starts from “[USR3]”, this means that we want to predict whether the learner with ID 3 can read the sentences that follow. Hence, “[USR3] It was a difficult period.” asks if the learner with ID 3 could read the sentence “It was a difficult period.”. The goal of the task is to predict 1 or 0, where 1 indicates that the learner could read the specified sentence, and 0 indicates that the learner was unable to do so. We fine-tuned the pre-trained BERT model in this manner using the “training” data shown in Figure 5. For the estimation, we used the Adam optimizer [13], in which the batch size was 32.

7.3 Results

Table 2 shows the predictive accuracies of all methods. The results showed that the prediction accuracy of the direct method was 64.4% and that of the alternative method was 54.4%, a difference of 10 points. This difference was significant at $p < 0.01$ in the Wilcoxon test. This result indicates that estimating the difficulty of atypical meanings of words from those of typical words is a challenging task.

Table 2: Predictive Accuracies of the Dashed Area

Base Method	Accuracy
IRT (ability - diffcl. of typical word)	0.544
IRT (ability - diffcl. of atypical word)	0.644
OURS (bert-large-cased)	0.674 (**)
OURS (bert-base-cased)	0.688 (**)
OURS (bert-based-uncased)	0.655
OURS (roberta-base)	0.681 (**)
OURS (albert-base-v2)	0.671 (*)

Our model achieved the best performance among the listed models. The name in () indicates the pre-trained model used for the experiments. In particular, our model significantly outperformed the IRT models with the best accuracy. This result was also statistically significant using the Wilcoxon test ($p < 0.01$). This is denoted by (**) in Table 2. As BERT considers the semantics of the question, this result suggests that the traits of each learner test-taker was captured via the embeddings of the [USR] tokens during the fine-tuning.

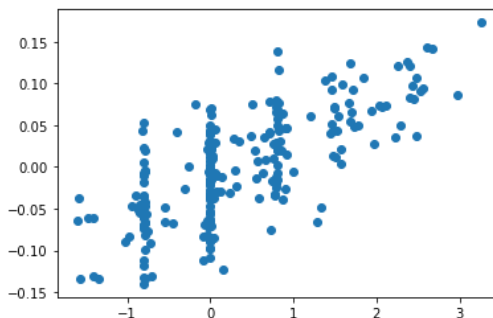
Table 2 showed the best performance for **bert-base-cased**. **bert-base-cased** achieved a performance better than **bert-large-cased**. The reasons of this are presumably as follows. Although the word embeddings of Transformer models are trained by many examples in their pre-trained corpus, the word embedding vectors of learner tokens, which represent learners’ characteristics, such as their abilities, were trained solely on a relatively small training data in the fine-tuning phase. Obviously, no learner tokens appeared in the pre-trained corpus. Hence, it could be possible that **bert-large-cased** has too many parameters to be tuned using the small training data in fine-tuning compared with **bert-base-cased**.

Table 2 also shows that a model must be *cased* to achieve a good accuracy, considering that **roberta-base** is cased whereas **albert-base-v2** is uncased. This is presumably because the model needs to recognize the start of a sentence, which starts with a capitalized word, since each question consists of a short sentence in this experimental setting.

8. EXTRACTING ABILITY VALUES

As stated above, our method handles learners as tokens. Transformer-based methods internally use “word embeddings” that represents the meaning of the tokens in the form of vectors. Hence, by obtaining the word embeddings of the learner tokens that we introduced, it was possible to analyze learners’ characteristics, such as their language abilities.

As word embeddings are typically multi-dimensional, dimension reduction methods such as principal component analysis (PCA), can be used to obtain abilities from the embeddings of learner tokens. Figure 6 shows the plot of a PCA of the ability parameters of test-takers of the vocabulary test dataset against the first principal component scores of each token of the token embeddings in the case of **bert-large-cased** in Table 2. A clear correlation can be observed between the two. The correlation coefficient was 0.72, and was statistically significant ($p < 0.01$). In this manner, learners’ abilities can be obtained through PCA of the test-taker tokens in our method, which means that our method is equipped

**Figure 6: Relationship between IRT ability parameters estimated by pyirt (horizontal) and the first principal component of the learner token embeddings (vertical)**

with the interpretability of IRT models.

In Figure 6, we used pyirt for estimating the ability parameters. To check the correlation using IRT software other than pyirt, following a standard textbook for educational psychology [21], we also conducted experiments using the R “ltm” package, which was developed completely independently of pyirt. Unlike pyirt, which uses MMLE, ltm uses the expected a posteriori (EAP) method for parameter estimation. Again, a statistically significant correlation was observed: the Pearson’s correlation was also 0.72, ($p < 0.01$).

9. CONCLUSION

In this study, we tackled the task of predicting whether language learners know the atypical meanings of a word and developed an evaluation dataset for this purpose. We proposed a simple method for applying MLMs to the aforementioned personalized prediction problem by introducing tokens that represent learners. The prediction performance of the proposed method was superior to that of IRT by a statistically significant margin. We also showed that, with the proposed method, one can easily obtain learners’ ability values using the first principal component scores of token embeddings. This result indicates that our method is highly interpretable and, hence, suitable for educational use.

The learner token embeddings that we introduced are multi-dimensional. While we showed that the first principal component score significantly correlated with the test-taker’s ability parameter, the other components may encode the learner’s other types of ability. In IRT, there is a similar idea to model the learner’s ability as a multidimensional vector, called “multidimensional IRT”. Our future work is to compare the other principal components of learner token embeddings with multidimensional IRT.

10. ACKNOWLEDGMENTS

This work was supported by JST ACT-X Grant Number JPMJAX2006 and JSPS KAKENHI Grant Number 18K18118, Japan. We used the ABCI infrastructure of AIST and the miniRaiden system of RIKEN for the computational resources. We thank Assoc. Prof. Joy Taniguchi and other members of the Shizuoka Institute of Science and Technology for their cooperation in preparing the dataset. We appreciate the anonymous reviewers’ valuable comments.

11. REFERENCES

- [1] V. Abou-Khalil, B. Flanagan, and H. Ogata. Personal Vocabulary Recommendation to Support Real Life Needs. In I. Roll, D. McNamara, S. Sosnovsky, R. Luckin, and V. Dimitrova, editors, *Artificial Intelligence in Education*, Lecture Notes in Computer Science, pages 18–23, Cham, 2021. Springer International Publishing.
- [2] F. B. Baker. *Item Response Theory : Parameter Estimation Techniques, Second Edition*. CRC Press, July 2004.
- [3] D. Beglar and P. Nation. A vocabulary size test. *The Language Teacher*, 31(7):9–13, 2007.
- [4] B. Choffin, F. Popineau, Y. Bourda, and J.-J. Vie. Das3h: modeling student learning and forgetting for optimally scheduling distributed practice of skills. In *Proc. of EDM*, 2019.
- [5] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proc. of NAACL*, 2019.
- [6] Y. Ehara. Building an English Vocabulary Knowledge Dataset of Japanese English-as-a-Second-Language Learners Using Crowdsourcing. In *Proc. of LREC*, May 2018.
- [7] Y. Ehara. Lurat: a lightweight unsupervised automatic readability assessment toolkit for second language learners. In *Proc. of ICTAI*, pages 806–814, 2021.
- [8] Y. Ehara, Y. Baba, M. Utiyama, and E. Sumita. Assessing Translation Ability through Vocabulary Ability Assessment. In *Proc. of IJCAI*, 2016.
- [9] Y. Ehara, Y. Miyao, H. Oiwa, I. Sato, and H. Nakagawa. Formalizing Word Sampling for Vocabulary Prediction as Graph-based Active Learning. In *Proc. of EMNLP*, pages 1374–1384, 2014.
- [10] Y. Ehara, I. Sato, H. Oiwa, and H. Nakagawa. Mining Words in the Minds of Second Language Learners: Learner-Specific Word Difficulty. In *Proceedings of COLING 2012*, pages 799–814, Mumbai, India, Dec. 2012. The COLING 2012 Organizing Committee.
- [11] Y. Ehara, N. Shimizu, T. Ninomiya, and H. Nakagawa. Personalized Reading Support for Second-language Web Documents. *ACM Trans. Intell. Syst. Technol.*, 4(2):31:1–31:19, Apr. 2013.
- [12] C. Fellbaum. Wordnet. In *Theory and applications of ontology: computer applications*, pages 231–243. Springer, 2010.
- [13] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *Proc. of ICLR*, 2015.
- [14] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut. ALBERT: A lite BERT for self-supervised learning of language representations. *CoRR*, abs/1909.11942, 2019.
- [15] B. Laufer and G. C. Ravenhorst-Kalovski. Lexical Threshold Revisited: Lexical Text Coverage, Learners’ Vocabulary Size and Reading Comprehension. *Reading in a Foreign Language*, 22(1):15–30, Apr. 2010.
- [16] J. Lee and C. Y. Yeung. Automatic prediction of vocabulary knowledge for learners of Chinese as a foreign language. In *2018 2nd International Conference on Natural Language and Speech Processing (ICNLSP)*, pages 1–4, Apr. 2018.
- [17] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019.
- [18] M. Maddela and W. Xu. A word-complexity lexicon and a neural readability ranking model for lexical simplification. In *Proc. of EMNLP*, pages 3749–3760, Oct.-Nov. 2018.
- [19] I. Nation. How Large a Vocabulary is Needed For Reading and Listening? *Canadian Modern Language Review*, 63(1):59–82, Oct. 2006.
- [20] I. S. P. Nation and R. Waring. *Teaching Extensive Reading in Another Language*. Routledge, Nov. 2019. Google-Books-ID: xRu_DwAAQBAJ.
- [21] I. Paek and K. Cole. *Using R for item response theory model applications*. Routledge, 2019.
- [22] B. Settles. Data for the 2018 Duolingo Shared Task on Second Language Acquisition Modeling (SLAM), 2018.
- [23] L. Sha, M. Rakovic, A. Whitelock-Wainwright, D. Carroll, V. M. Yew, D. Gasevic, and G. Chen. Assessing algorithmic fairness in automatic classifiers of educational forum posts. In *Proc. of AIED*, pages 381–394. Springer, 2021.
- [24] J. T. Shen, M. Yamashita, E. Prihar, N. Heffernan, X. Wu, S. McGrew, and D. Lee. Classifying math knowledge components via task-adaptive pre-trained bert. In *Proc. of AIED*, pages 408–419. Springer, 2021.
- [25] S. Xu, G. Xu, P. Jia, W. Ding, Z. Wu, and Z. Liu. Automatic task requirements writing evaluation via machine reading comprehension. In *Proc. of AIED*, pages 446–458. Springer, 2021.
- [26] S. M. Yimam, C. Biemann, S. Malmasi, G. Paetzold, L. Specia, S. Štajner, A. Tack, and M. Zampieri. A Report on the Complex Word Identification Shared Task 2018. In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 66–78, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

APPENDIX

A. DATASETS

The dataset used in this paper will be publicly available. Details of the dataset will be available at <http://yoebara.com/> or <http://readability.jp/>. Some previous datasets such as [6] are available at <http://yoebara.com/>.

B. DISCUSSION

In this paper, we have made two important suggestions. The first is to introduce learner tokens to apply Transformer models to the personalized prediction task. The second is that the learner ability can be extracted from the first principal component of the learner token embedding vectors.

A research question that directly follows from this result is: Is it always possible to extract learner ability from the Transformer models? We provide our views on this topic.

An important point of the experimental settings shown in Figure 5 is that all learner tokens are trained using the same 70 test questions. (Strictly speaking, as for the 100 learners in Figure 5, their learner tokens were trained without test questions for atypical words.) The word embeddings, other than the learner token embeddings, were already trained using the pre-trained model. Therefore, although words other than the learner token in a sentence had a strong influence on the training of learner token embeddings, they were all trained in a similar way except for the response of the learner to the test question text: correct/incorrect. Hence, it is natural that learner token embeddings mainly reflect the response of the learner to each question text.

Hence, in a setting in which each learner responds to completely different test question texts, it is expected that extracting the learner's ability values using the first principal score of the learner token embedding vectors will be difficult. This setting can also be seen in the case where the matrix Figure 5 is sparse because each column, i.e., each test question, was filled by a small number of learners.

C. SCATTER PLOTS

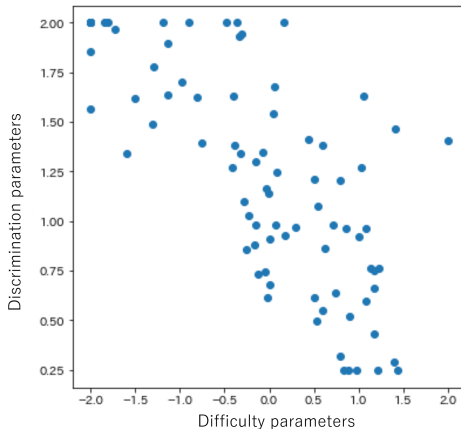


Figure 7: The horizontal axis shows the discrimination parameters and the vertical axis shows the difficulty parameters.

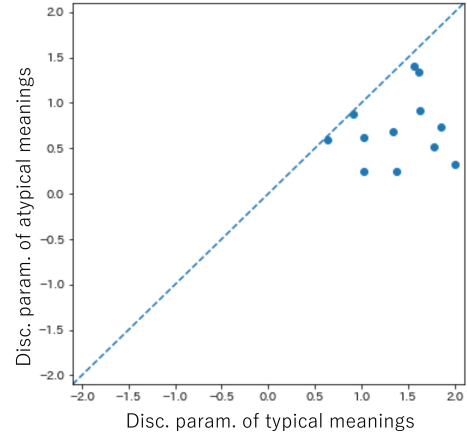


Figure 8: The horizontal axis shows the discrimination parameter of the typical meanings and the vertical axis shows the discrimination parameter of atypical meaning.

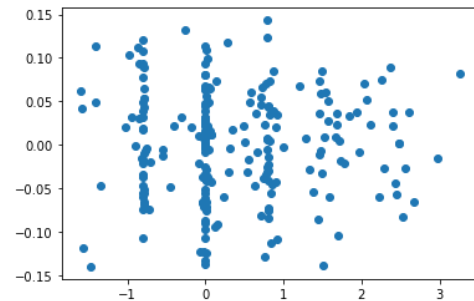


Figure 9: Relationship between the learner ability parameter estimated by the pyirt software (horizontal) and the second principal component of the learner token embeddings (vertical)

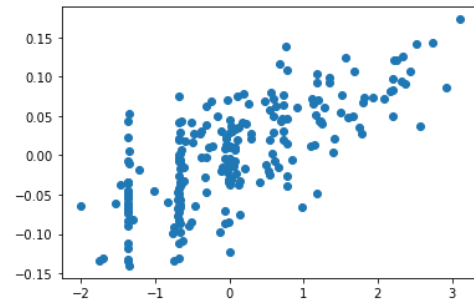


Figure 10: Relationship between IRT ability parameters estimated by ltm on the R language (horizontal) and the first principal component of the learner token embeddings (vertical)

Using community-based problems to increase motivation in a data science virtual internship

Jillian C. Johnson
Institute for Intelligent Systems
The University of Memphis
jillian.johnson@memphis.edu

Andrew M. Olney
Institute for Intelligent Systems
The University of Memphis
aolney@memphis.edu

ABSTRACT

Typical data science instruction uses generic datasets like survival rates on the Titanic, which may not be motivating for students. Will introducing real-life data science problems fill this motivational deficit? To analyze this question, we contrasted learning with generic datasets and artificial problems (Phase 1) with a community-sourced dataset and authentic problems (Phase 2) in the context of an 8-week virtual internship. Retrospective survey questions indicated interns experienced increased motivation in Phase 2. Additionally, analysis of intern discourse using Linguistic Inquiry and Word Count (LIWC) indicated a significant difference in linguistic measures between the two phases. Phase 1 had significantly greater measures of pronouns with a small-medium effect size, 2nd person words with a medium-large effect size, positive emotion with a medium effect size, inter-rogations with a medium-large effect size, question marks with a medium-large effect size, risk with a medium-large effect size, and causal words with a medium effect size. These results in conjunction with a retrospective survey suggest that phase 1 had more questions asked, more causal relationships defined, and included linguistic features of success and failure. Results from Phase 2 indicated that community-sourced data and problems may increase motivation for learning data science.

Keywords

Virtual internship, motivation, service-learning, language analysis

1. INTRODUCTION

Data science curriculum development is challenging due to prerequisites in statistics, programming, and machine learning [35]. Dataset complexity is another challenge: while educators acknowledge that data science embraces messy data, typical practice is to use sanitized or “canned” datasets to demonstrate a particular approach [3][10]. For example, the UCI Machine Learning Repository, a popular source of datasets, lists datasets on irises, adult income in 1996, and the geographic origin of wines as its top three downloaded datasets [9].

The practice of using canned datasets illustrates the pedagogical tension between keeping intrinsic cognitive load low [33] without sacrificing learning opportunities to develop key data science skills

J. Johnson and A. Olney. Using community-based problems to increase motivation in a data science virtual internship. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 500–507, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853167>

for working with messy data. Cognitive load theorists have proposed that motivation is particularly important for learning complex skills over time, because motivation causes learners to invest in germane cognitive load [20].

In the context of learning data science, dataset manipulation is a potential avenue for increasing motivation. Personalization has been used in previous research to increase motivation for learning [8]. For example, personalization might entail allowing the learner to choose the dataset or matching a dataset based on the learner’s preference profile. However, this type of data personalization can be challenging because the data in question may not be accessible or suitable for advancing a learning goal.

The alternative explored in the present study is to use datasets and problems sourced from community partners. In the framework of self-determination theory [26], this approach should build intrinsic motivation through the constructs of relatedness (by working on a problem of concern in their community), autonomy (by deciding how to address the problem of concern rather than being told to perform a specific analysis), and potentially competence (by making progress on the problem and so increasing self-efficacy in data science).

Within our research context of an 8-week data science virtual internship, we hypothesized that interns would experience increased motivation during the final phase of the internship in which they worked on community-based problems. To evaluate this hypothesis, we conducted retrospective surveys and analyzed the communications between interns for linguistic indicators of increased motivation, effort, confidence, competence, and emotion.

2. BACKGROUND

2.1 Internship & motivation

Traditional internships offer a markedly different context for learning compared to formal education. While formal education typically engages in prescriptive or rote learning, internships are grounded in real-world tasks that have material impacts on interns in terms of compensation and future employment opportunities. As such, internships have substantial potential to enhance motivation around learning in a way that parallels, and perhaps even surpasses, project-based learning in formal education [2].

The motivational impacts of internships have been found across the literature. [15] worked with IT interns to consider the roles of tasks, learners, and mentors in a project-based intern program. It was found that mentors increased the learners’ successful expectancies and therefore increased learners’ self-efficacy. Because of this, it was hypothesized that mentoring increases the learner’s self-determination and subsequently their motivation. [19] a study based on

Model

Next create a simple linear regression model.

Start by importing `sklearn.linear_model` and `numpy`.

```
[1]: import sklearn.linear_model as linear_model
import numpy as np

#<small>xml xmlns="https://developers.google.com/blockly/xml"><variables><variable id="f+Hl;Yx;ZBIEQV8ItP0">linear_model</variable><variable id="YjnR+H75hTgk vKFR0Dx">np</variable></variables></small>
<small>...</small>

Create the model.

[2]: lm = linear_model.LinearRegression()

#<small>xml xmlns="https://developers.google.com/blockly/xml"><variables><variable id="F]qI47x/*w|PWfPQU-LZ">lm</variable><variable id="f+Hl;Yx;ZBIEQV8ItP0">linear_model</variable></variables></small>
<small>...</small>

Train the model.

[12]: lm.fit(dataframe[['Height']], dataframe[['Volume']])

#<small>xml xmlns="https://developers.google.com/blockly/xml"><variables><variable id="F]qI47x/*w|PWfPQU-LZ">lm</variable><variable id="85p-XuLdIZ.88nd96oa8">dataframe</variable></variables></small>
<small>...</small>

[12]: LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)

Print out the slope and intercept.

[15]: print('Slope:' + str(lm.coef_))
print('Intercept:' + str(lm.intercept_))

#<small>xml xmlns="https://developers.google.com/blockly/xml"><variables><variable id="F]qI47x/*w|PWfPQU-LZ">lm</variable></variables><block type="text_print" id="j58cP79*W9VvZAHYKq?></block></small>
<small>...</small>

Slope: [[1.54334975]]
Intercept: [-87.12361354]
```

Figure 1 Jupyter Notebook Lesson on Simple Linear Regression

167 college interns working in retail, it was discovered that emotional sharing is positively related to learning and mentoring. Research in cognitive neuroscience leads us to believe that emotions activate neural circuits which engage sensory systems that increase attention and motivate perceptual processing [17]. Positive emotion was hypothesized to relate to motivation in that by increasing these perceptual factors, learners would be more compelled to learn.

Similar results have been found in virtual internships [13]. Virtual internships are becoming a popular alternative to in-person structures due to the Covid-19 Pandemic, and they have the potential to offer a unique educational strategy. Research has shown that learning efforts outside of traditional classrooms are needed to address systemic disparities within education [30]. In [25] it was seen that virtual internship programs provide a quality opportunity for non-traditional students to participate in practical experiences regardless of their physical location and other obstacles.

The virtual internship discussed in this paper follows through on goals to help students during the pandemic. This includes paying learners, offering them loaner laptops, and creating a pedagogy that motivates students when working with a local community partner in need.

[18] showed that service-learning increased civic skills, problem-solving, and motivation. This leads to the idea that service-learning internships have the potential to enhance motivation, especially if the service is aligned with the intern's beliefs and values. Service-

learning gained attention in the 1970s [29] and has become increasingly popular in engineering [4] as well as data science, as evidenced by such programs as Data Science for the Social Good.

2.2 Research Context

We designed an 8-week data science internship with two phases. Phase 1 of the internship was an educational data science boot camp that consisted of Jupyterlab Python notebooks¹ with a Blockly plugin (Anonymous) so that students could solve data science problems with a block-based programming language. Learners were split into pods which was a distributed team of students ranging from 3-4 people including a student mentor. The notebooks consisted of materials that span across data science topics, such as cleaning data, Random forests, regression trees, and cross-validation. To cover these topics according to a fixed schedule, Phase 1 used common generic datasets with artificial problems.

(i.e., problems proscribed by the learning materials). Each topic was covered by introducing it as a worked example in the morning followed by problem-solving in the afternoon. At the end of the day, they would engage in peer grading and review, which would culminate in a group discussion led by a faculty member. In addition to the notebooks, students were provided a reference manual that abstracted key steps from the notebooks, based on an observation that interns sometimes struggled with learning transfer (Anonymous). Further, Phase 1 implemented a problem-based learning environment, with mentors to help get through the questions. This style of

¹ <https://github.com/memphis-iis/datawhys-content-notebooks>

Zip codes served

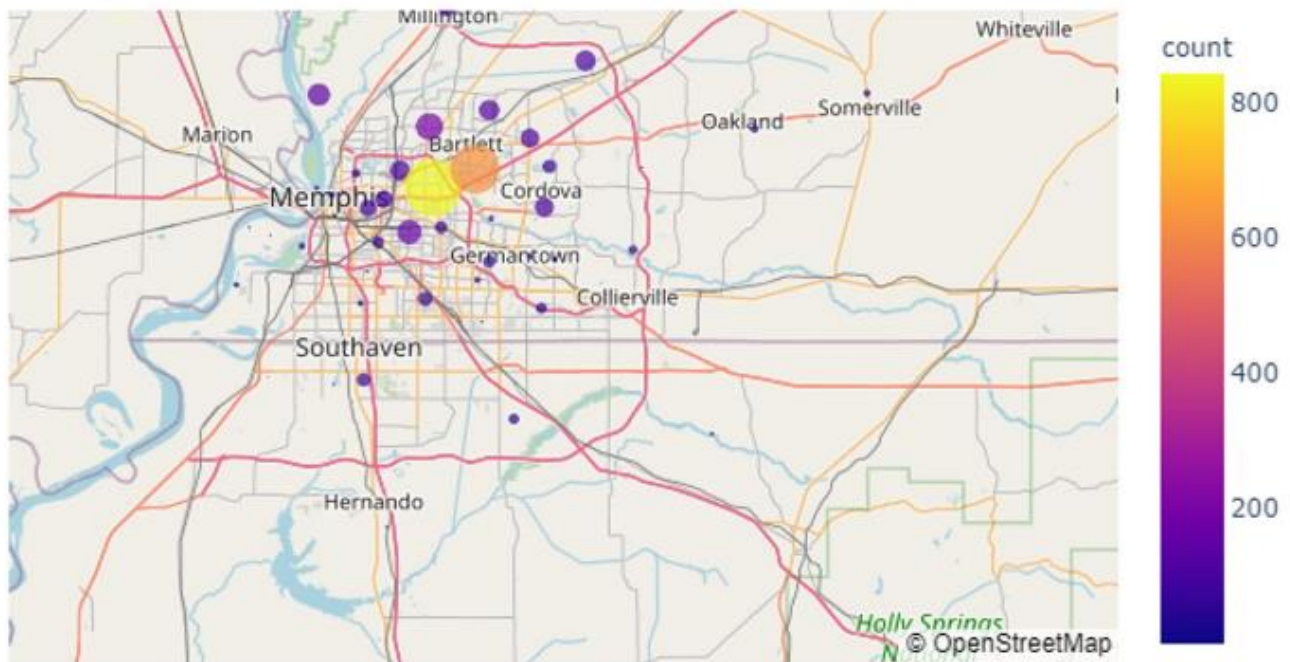


Figure 2 Intern Visualization for A Betor Way Project

learning has been seen to increase intrinsic motivation [5] [16]. This is a type of instructional design suggested by [30] such that students learn from failure, and that they cannot fail in this environment due to the time allowed to rework, fix, and learn from their mistakes. This is motivating due to the lack of pressure on the student to be correct.

In Phase 2 of the internship, students were regrouped into two teams such that they could work on a respective community-partner project. The projects consisted of harm-reduction data from a clean needle and Narcan mutual aid group (A Betor Way), and event data from a group that helps battered women get into safe homes and supplies resources (Restoration Time Family Youth Services). Both community partners are locally based. Before Phase 2, each partner presented to the interns and faculty for 10 minutes and took questions about their work and data. After the presentations, each intern rank-ordered their preferences of which project they wanted to be on, which was considered and distributed based on their preferences and the number of students in each group. Each team worked with 1-2 faculty members that they met with on a regular basis (e.g., 2-3 times a day). The faculty members provided guidance and helped set goals and schedules. Each team was also provided with a captain that was nominally for coordinating within team members. An important factor of Phase 2 is that it keys in on reinvention and reconfiguration during a learning task, which has been seen to add meaning to the learning experience [6]. In an experiential activity, such as a real-world authentic problem, motivation to learn is cultivated [24]. This can be seen in a study conducted over five forms of experiential learning, which reported a high self-perception of learning in a service-learning setting [7].

2.3 LIWC

Linguistic Inquiry and Word Count (LIWC) calculates linguistic measures using a dictionary-based approach that counts words in

the given text according to the categories in which they belong [24]. For example, if you use the word “sad” it will be counted as a target word that matches a dictionary word for the category of *negative emotion words*. The dictionary is composed of 94 categories and almost 6,400 dictionary words like this. The full comprehensive text can be found in [24]. The 2015 version of the software, which was used in this study, supports the languages of Spanish, German, Dutch, Norwegian, Italian, and Portuguese.

The usage of LIWC in discourse spans many topics in education such as understanding learners’ difficulties, discovering motivational insights to learning styles, and analyzing the sentiment of experts and non-experts over time. In [11] it was seen that various language modeling tactics, including LIWC, lead to the understanding that researchers could interpret learners’ needs and difficulties in learning. This insight was used to change the materials and course settings such that the learner experience was increased [11]. LIWC was also used in [36] to develop research around learning styles while classifying the motives behind them. This study found that LIWC could be used to identify the motives for different students’ needs and what learning styles could be used to facilitate them. Further showing the application of LIWC in educational settings, [1] studied the emotional states of learners in an online Stack Overflow learning community. They found that learners were more analytical and less authentic over time, meaning learners progressed in their learning capabilities but became less honest in their posts. It was also found that the clout (e.g., confidence) levels of non-experts decreased overtime in their question-and-answer posts, while experts only had a decrease in clout for their answer posts – not the question posts. This indicates that learners became less confident in their materials over time, while experts doubted their answers but had strong questions. LIWC has been used across varying processes, tasks, and materials to understand the social reality of learner discourse.

In the present study, LIWC2015 was used to analyze the change in discourse between Phase 1 and 2 as an indicator of motivation, emotional resonance, confidence, competence, and effort. These constructs were analyzed using LIWC with data extracted from Discord, an online messaging and communication platform that learners used to communicate with each other, their mentors, and faculty. It is important to note that from a psychological perspective, style words such as pronouns and 2nd person reflects how people are communicating, while content words such as the ones that fall under the positive and negative emotion category convey what people are saying [34]. In terms of what people are saying, different variables, such as word count, can be used to determine the characteristics of a speaker, such as effort. In this way, the discussed constructs can be operationalized in terms of the LIWC variables. For example, affective indicators of emotional tone, affect, positive emotion, and negative emotion could be used to represent emotional tendencies. The reason for looking at this construct is the expectation that when students have a moral responsibility to help a community partner in need, they will have a larger emotional resonance with the task and therefore be more motivated.

3. METHODS

3.1 Participants

There were 10 participants in this study, 5 men and 5 women. For the A Betor Way project, there were 3 men and 2 women. For the Restoration Time Family Youth Services project, there were 2 men and 3 women. All but one intern was from LeMoyne Owen College, a private Historically Black College and University (HBCU) in Memphis, TN. The HBCU participants came from a variety of majors, and the remaining participant was an incoming data science graduate student. Three participants were mentors. One was an incoming graduate student, and the other two were former interns.

3.2 Materials and Procedure

Data for LIWC analysis were collected through mining the conversations students were having over the messaging platform Discord. Channels for each pod were created, a help channel, a general chat channel, and two channels for each community-partner project. The text logs from Phase 1 were collected using the pod channels, and the text logs from Phase 2 were collected using the community-partner channels. Voice channels for general, help, each team, and each project were also created, but data from these sources has been excluded because we did not save this method of communication.

Discord chat logs were exported as JSON files using the Discord Chat Exporter². Then, we used a Python script to collect word counts and length of words per post by each student by afternoon and morning notebook. We aggregated the posts from each student into a single text for Phase 1 and again for Phase 2. The resulting texts were put into an Excel file and used for LIWC analysis with each student and all the words they used per phase as a datapoint.

Survey items were distributed to students retrospectively (e.g., at the end of phase 2) over Google Forms. The items were constructed by centering the constructs of motivation and affiliation. Questions used a 5-point Likert scale, designating no influence in the question asked to a very strong influence. Two of the questions, the last two

rows in Table 2, used the number three option as a designator for no influence whatsoever.

4. RESULTS

Table 1 LIWC measures with significant changes across phases

Measure	Phase 1		Phase 2		<i>p</i>	<i>d</i>
	<i>M</i>	<i>SD</i>	<i>M</i>	<i>SD</i>		
pronoun	59.00	48.85	40.90	55.86	.048	.34
you	13.20	16.22	4.75	7.42	.027	.67
interrog	8.50	5.70	4.84	5.44	.048	.66
posemo	16.10	13.24	9.40	13.61	.037	.50
cause	9.81	7.91	5.55	8.30	.009	.53
risk	3.58	3.36	1.75	2.69	.027	.60
Qmark	8.68	8.16	4.44	6.63	.019	.57

4.1 LIWC

All measures reported by LIWC were analyzed using the Wilcoxon signed ranks test, but due to space constraints, we only report significant results in Table 1. The indicated p-values are not corrected for significance due to chance (type 1 error), which is expected when performing 88 tests at once. When we corrected the p-values using the strict Holm-Bonferroni method and less strict Benjamini-Hochberg method, no significant differences were found. Post-hoc power analysis using GPower 3 revealed that the design was underpowered, with power .28 to find a medium effect with $\alpha = .05$.

All measures in Table 1 significantly decreased from Phase 1 to Phase 2. Personal pronouns are “I, them, her itself” and include ‘you’. This measure had small-medium effect size, whereas the other measures had a medium or medium-large effect size You measures 2nd person words such as “you, your, yourself”. Interrog means interrogatives such as “how, when, what”. Posemo means positive emotion and includes words such as “love, nice, sweet”. Cause means causation words, such as “before, effect”. Risk means words associated with risk such as “danger, doubt”. Finally, Qmark means the number of question marks used.

4.2 Survey Results

Only 5 interns responded to the survey. In the survey results, it was found that students self-reported higher motivation when working on the community projects in Phase 2. Table 2 shows the survey questions and their respective scores. The scores across all questions indicate that the interns perceived an increase in motivation during Phase 2. Additionally, interns developed a sense of understanding that data science can affect their local communities and it helped develop a sense of connection to their communities. Overall,

² by Github user Tyrrrz (<https://github.com/Tyrrrz/DiscordChatExporter>).

the outcomes of this survey were very positive, but limitations exist in the lack of power present with such a low N.

Table 2 Retrospective motivational survey of interns

Question	<i>M</i>	<i>SD</i>
How did working with a community partner increase your motivation?	5.00	.00
How did working on your team’s project increase your connection to the community?	4.00	.49
How did knowing that your project was with community-based data increase your efforts?	5.00	.00
To what degree did working with a community partner influence your perspective on data science’s power to have a local impact?	4.43	.19
To what degree did your interest in social justice work change since the beginning of the community project	4.20	.44

5. DISCUSSION

5.1 Interpretation

Students reflexive posting indicates that they wrote more questions, explained the processes, took more risks, and had a more positive attitude in Phase 1. Nothing resulted in an increase in Phase 2, which was contrary to our results. The present study expected to primarily see an increase in motivation, and secondarily see an increase in effort, competence, confidence, and emotion. The only construct in the LIWC analysis that matches what we were expecting was positive emotion, but the results came out in the other direction. However, our primary hypothesis of motivation comes out correct in the retrospective survey results.

Along with questions being asked, we can interpret the causal words, 2nd pronoun usage, and pronoun usage. These categories indicate the prevalence of explanations by means of needing to indicate causal relationships, direct others on what to do, and refer to objects and people. Additionally, since “pronoun” and “you” words both decreased, this means no other subtypes of pronouns changed. Since “you” is nested in “pronoun”, this implies that the change was all in “you”. In other words, the “pronoun” result is entirely dependent on “you”. In [31] pronouns were negatively related to relationship quality. This means that since “you” decreased, an improvement in relationship quality changed from phase 1 to phase 2. However, this could just be due to time, not to the phase 2 activity.

The results of interrog and Qmark indicate the construct of questions, so it can be said more questions were asked in Phase 1. This makes sense because there is a back-and-forth dialogue between learners and mentors – both must ask each other questions. In Phase 2, questions seemed to be one direction, as in learners and mentors were asking questions that a faculty member could answer with or without another question. Here, it is important to consider that faculty posts were not measured, so if they responded back with a question, it was not reported. Additionally, it was found in [27] that lower-status language is more self-focused and tentative, while high-status language speaks more often and freely makes statements. The use of first-person plural usage correlated with higher rank, but the opposite pattern was found for question marks

compared to lower-ranked members of a crew. A reduction of question marks could indicate an increased sense of status and therefore, self-efficacy.

Because causal words went down, the language can be said to have been less complex over time as the interns had more knowledge and didn’t have to explain themselves as much when talking to each other. This can be related to literature from [14] that such discussions are the most complex part of an article because results must be integrated and differentiated from past findings. It can also be related to the literature of [12] which studied how prepositions signal the speaker is providing more complex and concrete information about a topic, showing that words greater than six letters are also indicative of more complex language. These causal relationships show that there is a complexity to the conversation which does not occur in Phase 2 possibly due to the nuanced form of problem-based learning.

The results of risk and positive emotion are a bit fuzzier, but they seem to reflect a dichotomy of failure and success. Risk words include a language of doubt and tentability. This is a bit strange to see in Phase 1, but it starts to make sense when contextualized to the failure literature of [30]. When students are in an environment where they have the option to fail, doubt occurs, and a sense of risk is invoked due to the consequential fear of getting problems wrong. Positive emotion reflects the opposite of this – the joy of getting answers correct. Since there are no problem sets in Phase 2, only a project to complete, there are less iterations of the failure and success dichotomy, which results in fewer positive emotion and risk words. Additionally, since positive emotion words went down, this could indicate there was less agreement in Phase 2 than Phase 1. The task was more difficult and had multiple solutions in Phase 2. Since risk also went down, this could mean that concerns went down. This would make sense because the learners had an increased ability at this point. Lowering of these concerns could indicate increased self-efficacy.

In the rest of this section, the nuances of each result will be discussed:

Pronoun means total pronouns used. This is represented by the words “I, them, her, itself”. It was found that the use of pronoun words dropped in Phase 2, which could be because there was more direct mentoring in Phase 1. This is exemplified in the sentence “You’re gonna set one of them to Import: “pandas” – as – “pd”. This sentence shows the use of variable pronouns directed at another person in need of assistance. Compare this to the sentence: “Sorry, I got super dugged dow with a dumb error on the scatter matrix, but I’m done now. I’m trying to add a new/return-dependent color to the Narcan Given histogram.” This shows the use of primarily one type of pronoun – personal. It is possible that in Phase 1 lots of different pronouns were used, indicating acts of mentoring and calls for help of assistance, while Phase 2 used a smaller category of pronouns to update the chat on what they were doing on the project.

You means 2nd person words used. This is represented by the words “you, your, yourself”. IT was found that the amount of you were dropped in Phase 2, which could indicate there was more instruction in Phase 1. This is similar to the thought pattern behind the pronoun decrease – there was more variability in pronouns, especially 2nd person words, in the first phase but the instruction of this type was lacking in Phase 1. However, we do not see an increase in 1st person words in Phase 2, which means if there was an increase in these types of words in Phase 1 it did not happen at any significant level.

Interrog means interrogatives, and Qmark means question marks. These words are represented by “how, when, what” and “?” respectively. They will be discussed together due to the correlation of the categories and ideological similarities in what they represent – questions. There were more interrogatives and question marks in Phase 1 than in Phase 2, and they both had about the same number in both phases. This could be because interrogatives and question marks are linked in their representation of questions, which makes sense for these to be more prominent in Phase 1 than Phase 2 due to the direct problem-solving nature of tasks in Phase 1. In Phase 2, there seemed to be a culture of putting question marks behind things they were trying to verify, such as: “I guess we have to get dummies for all the non-numerical stuff, right?”, whereas in Phase 1 there were more questions from both interns and mentors, such as the question: “What kind of error are you getting?”. Mentors needed to ask questions to assist, as well as interns asking the questions to get answered. This two-ended need for questions could account for the almost double amount of question marks and interrogatives in Phase 1.

Cause means causation words, such as “before, effect”. These words were higher in Phase 1 than in Phase 2, which could be because of a need for understanding the problems in Phase 1, and the mentors giving answers in this fashion. Take the example sentence “Ok, it’s because the last 3 freestyle blocks go outside the main block, like this: [screenshot]”. It shows a mentor answering a question in a causal style. Examples like this are plentiful in Phase 1, but they are lacking in Phase 2 due to less of a need to understand what is happening and more of a need to update the channel on what they are accomplishing. The explanation of how or why they are doing such a thing is not there because everyone already understands the underlying mechanisms.

Risk means risk in the LIWC dictionary, and it accounts for words such as “danger, doubt”. This could be because of learners’ hesitancy in what the materials they are learning. It represents a tenability in answering problems because there is an underlying fear of getting the questions wrong.

Posemo means positive emotion, which is represented by the words “love, nice, sweet”. These words were more prominent in Phase 1 than in Phase 2, contrary to our hypothesis. This could be because learners were more likely to express gratitude for answers in Phase 1, and there were not that many questions in Phase 2. Additionally, it could represent the happiness one feels when getting an answer correct or getting the solution they need. It could also represent the gratitude learners had to their mentors for helping them. In this sense, the linguistic feature of positive emotion represents the other opposite of risk, meaning these two categories might have a relationship in the problem-based learning context. Overall, these results can be interpreted as more questions being asked, and more causal relationships being defined.

5.2 Limitations

The study has several limitations. The study was nonexperimental with low sample size, reducing our power to find an effect or claim causality. The order of the phases was not counterbalanced, by necessity, so it is possible that some changes in discourse are due to maturation and not the community project in Phase 2. The analysis done in LIWC uses a dictionary approach – words are matched based on predefined categories instead of studying word relationships and their contextual clues [22]. Finally, the survey questions were asked retrospectively, so there was no baseline to measure change, and the participants may have had bias responses due to the retrospective phrasing of the questions.

5.3 Future Research

In the future, more participants should be studied because the power of this research is very low for both LIWC and Survey measures. Motivational surveys should be distributed before and after the internship, instead of just after. Also, questions should be reworded and put on the same scale.

This study should also be used as a comparison metric for the same internship next year. By doing so, insights could be found on the similarity between years to see if any language changed over the course of a year’s development of the program.

Another future research point would be to include the 4 summary variables in the LIWC dictionary (e.g., analytic, clout, tone, and authentic) to see if there were any changes over the phases per percentile scoring. Word count could also be analyzed, as well as words per sentence if set up correctly with stopper marks.

Finally, it would be good to measure the language usage in a time series of days per week over the four weeks to see what changes happen per day. This would allow us to see the movement and variability of sentiment change.

5.4 Conclusions

In conclusion, this study shows that learners asked more questions, described more relationships, and were more positive in Phase 1. It was also found that students were more motivated in Phase 2.

It is important to look at the LIWC analysis between phases because it highlights the psychological underpinnings of learners. There is potential to discover constructs laying within the text. In the case of this study, those constructs were more questions asked, discussion of causal relationships, and the success and failure influences.

Even though our hypotheses of confidence, competence, and effort were not detected, we did find increases of motivation and discovered constructs that exist in Phase 1. This is valuable research because it suggests that in a problem-based learning environment with mentors more questions will be asked, more relationships will be discussed, learners will be willing to take more risks, and they will emotionally reap the rewards of getting things correct by taking those risks.

This study also resulted in a finding of our secondary hypothesis of an emotion change, but it occurred in the opposite direction than we were expecting. This result was attributed to the joy of success when solving problems, instead of the joy of working with a community partner. Although we did not see a linguistic increase in positivity, we did find that learners had an increased interest in social justice activities and were more motivated to complete the project. This means that despite the LIWC results, we can still say Phase 2 had an impact on learners.

The significance of this research lays in educational design such that project-based service-learning programs do increase motivation, and a problem-based environment induces the discussed constructs.

6. ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 1918751. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. Additionally, we would like to thank Dr. Andrew Tawfik and Linda Payne for helpful and engaging conversations.

7. REFERENCES

- [1] Adaji, I. and Olakanmi, O. 2019. Evolution of Emotions and Sentiments in an Online Learning Community. *International Workshop on Supporting Lifelong Learning*. (2019).
- [2] Barron, B., Schwartz, D., Vye, N., Moore, A., Petrosino, A., Zech, L. and Bransford, J. 2011. Doing with Understanding: Lessons From Research on Problem- and Project-Based Learning. *The Journal of Learning Sciences*. 7, 3–4 (2011), 271–311.
DOI:<https://doi.org/10.1080/10508406.1998.9672056>.
- [3] Baumer, B. 2014. A Data Science Course for Undergraduates: Thinking With Data. *The American Statistician*. 69, 4 (2014), 334–342.
DOI:<https://doi.org/10.48550/arXiv.1503.05570>.
- [4] Bielefeldt, A.R. and Swan, C.W. 2010. Measuring the Value Added from Service Learning in Project-Based Engineering Education. *International Journal of Engineering Education*. 26, 3 (2010), 535–546.
- [5] Blumenfeld, P., Soloway, E., Marx, R.W., Krajcik, J.S., Guzdial, M. and Palincsar, A. 2011. Motivating Project-Based Learning: Sustaining the Doing, Supporting the Learning. *Educational Psychologist*. 26, 3–4 (2011), 369–398.
DOI:<https://doi.org/10.1080/00461520.1991.9653139>.
- [6] Cam, P. Dewey’s Continuing Relevance to Thinking in Education. *Philosophical Reflections for Educators*.
- [7] Coker, J.S. and Porter, D.J. 2016. Student Motivations and Perception Across and Within Five Forms of Experiential Learning. *The Journal of General Education*. 65, 2 (2016), 138–156. DOI:<https://doi.org/10.5325/jgeneeduc.65.2.0138>.
- [8] Cordova, D.I. and Lepper, M.R. 1996. Intrinsic motivation and the process of learning: Beneficial effects of contextualization, personalization, and choice. *Journal of Educational Psychology*. 88, 4 (1996), 715–730.
DOI:<https://doi.org/10.1037/0022-0663.88.4.715>.
- [9] Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [<http://archive.ics.uci.edu/ml>]. Irvine, CA: University of California, School of Information and Computer Science.
- [10] Erickson, T., Finzer, B., Reichsman, F. and Wilkerson, M. 2018. Data Moves: One Key To Data Science At The School Level. *Proceedings of the International Conference on Teaching Statistics*. 10, (2018), 6.
- [11] Geng, S., Niu, B., Feng, Y. and Huang, M. 2020. Understanding the focal points and sentiment of learners in MOOC reviews: A machine learning and SC-LIWC-based approach. *British Journal of Educational Technology*. 5, 1 (2020), 1785–1803.
DOI:<https://doi.org/doi:10.1111/bjet.12999>.
- [12] Graesser, A.C., McNarma, D.S., Louwerse, M.M. and Zhiqiang, C. 2004. Coh-Metric: *Analysis of text on cohesion and language*. *Behavior Research Methods, Instruments & Computers*, 36, 2 (2004), 193–202. DOI:
<https://doi.org/10.3758/BF03195564>
- [13] Grangeia, T. de A.G., Jorge, B. de, Franci, D., Santos, T.M., Setubal, M.S.V., Schweller, M. and Carvalho-Filho, M.A. de 2016. Cognitive Load and Self-Determination Theories Applied to E-Learning: Impact on Students’ Participation and Academic Performance. *Plos One*. (2016).
DOI:<https://doi.org/10.1371/journal.pone.0152462>.
- [14] Hartley, J., Pennebaker, J.W., Fox, C. 2003. Abstracts, introductions and discussions: How far do they differ in style? *Scientometrics*. 57, 3 (2003), 389–398. DOI:
[10.1023/A:1025008802657](https://doi.org/10.1023/A:1025008802657)
- [15] Johari, A. and Bradshaw, A. 2008. Project-based learning in an internship program: A qualitative study of related roles and their motivational attributes. *Educational Technology Research and Development*. 56, 3 (2008), 329–359.
DOI:<https://doi.org/10.1007/s11423-006-9009-2>.
- [16] Land, S. and Hannafin, M. 1996. A conceptual framework for the development of theories-in-action with open learning environments. *Educational Technology Research and Development*. 44, 3 (1996), 37–53.
DOI:<https://doi.org/10.1007/BF02300424>.
- [17] Lang, P. and Bradley, M. 2010. Emotion and the motivational brain. *Biological Psychology*. 84, 3 (2010), 437–450.
DOI:<https://doi.org/10.1016/j.biopsycho.2009.10.007>.
- [18] Levesque-Bristol, C., Knapp, T.D. and Fisher, B.J. 2011. The Effectiveness of Service-Learning: It’s Not Always what you Think. *The Journal of Experiential Learning*. 33, 3 (2011), 208–224.
DOI:<https://doi.org/doi:10.1177/105382590113300302>.
- [19] Liu, Y., Xu, J., and Weitz, B.A. 2011. The Role of Emotional Expression and Mentoring in Internship Learning. *Academy of Management Learning*. 10, 1 (2011).
DOI:<https://doi.org/10.5465/amle.10.1.zqr94>.
- [20] Merriënbor, J.J.G. and Sweller, J. 2005. Cognitive Load Theory and Complex Learning: Recent Developments and Future Directions. *Educational Psychology Review*. 17, 2 (2005), 31. DOI:<https://doi.org/10.1007/s10648-005-3951-0>.
- [21] Molock, S. and Parchem, B. 2021. The impact of COVID-19 on college students from communities of color. *Journal of American College Health*. (2021).
DOI:<https://doi.org/10.1080/07448481.2020.1865380>.
- [22] Moore, R.L., Yen, C.J., and Powers, F.E., 2020. Exploring the relationship between clout and cognitive processing in MOOC discussion forums. *British Journal of Educational Technology*. 52, 1 (2020), 482–497. DOI:
<https://doi.org/10.1111/bjet.13033>
- [23] Payne, L., Tawfik, A. and Olney, A. 2021. Datawhys Phase 1: Problem Solving to Facilitate Data Science & STEM Learning Among Summer Interns. *International Journal of Designs for Learning*. 12, 3 (2021), 102–117.
DOI:<https://doi.org/10.14434/ijdl.v12i3.31555>.
- [24] Pennebaker, J.W., Boyd, R.L., Jordan, K. and Blackburn, K. The Development and Psychometric Properties of LIWC 2015. *The University of Texas, Austin*. 2015.
- [25] Ruggiero, D. and Boehm, J. 2016. Design and Development of a Learning Design Virtual Internship. *The International Review of Research in Open and Distributed Learning*. 17, 4 (2016). DOI:<https://doi.org/10.19173/irrodl.v17i4.2385>.
- [26] Ryan, R.M. and Deci, E. 2000. Self-Determination Theory and the Facilitation of Intrinsic Motivation, Social Development, and Well-being. *American Psychologist*. 55, 1 (2000), 68–78. DOI:<https://doi.org/10.1037/0003-066X.55.1.68>.
- [27] Sexton, J.B. and Helmreich, R.L. 2000. Analyzing cockpit communications: The links between language, performance, error, and workload. *Human Performance in Extreme Environments*, 5, 1 (2000), 63–68.

- [28] Sibthorp, J., Furman, N., Paisley, K., Gookin, J. and Schumann, S. 2011. Mechanisms of Learning Transfer in Adventure Education: Qualitative Results From the NOLS Survey. *Journal of Experiential Education*. 34, 2 (2011), 109–126. DOI:<https://doi.org/10.5193/JEE34.2.109>.
- [29] Sigmon, R. 1979. Service-Learning: Three Principles. *Synergist*. (1979).
- [30] Simpson, A. and Maltese, A. 2017. “Failure is a major component of learning anything”: The role of failure in the development of STEM professionals. *Journal of Science Education and Technology*. 26, 2 (2017), 223–237. DOI:<https://doi.org/10.1007/s10956-016-9674-9>.
- [31] Simmons, R. A., Chambless, D. L., Gordon, P.C. 2008. How do hostile and emotionally overinvolved relatives view relationships? What relatives’ pronoun use tells us. *Family Process*. 47, 3 (2008), 405–419. DOI: <https://doi.org/10.1111/j.1545-5300.2008.00261.x>
- [32] Stansbie, P. and Nash, R. 2013. Internship Design and Its Impact on Student Satisfaction and Intrinsic Motivation. *Journal of Hospitality & Tourism Education*. 25, 4 (2013), 157–168. DOI:<https://doi.org/10.1080/10963758.2013.850293>.
- [33] Sweller, J. 1994. Cognitive Load Theory, Learning Difficulty, and Instructional Design. *Learning and Instruction*. 4, (1994), 295–312. DOI:[https://doi.org/0959-4752\(94\)00010-7](https://doi.org/0959-4752(94)00010-7).
- [34] Tausczik, Y.R. and Pennebaker, J.W. 2009. The psychological meaning of words: LIWC and computerized text analysis methods. *Journal of Language and Social Psychology*. 29, (2009). DOI:<https://doi.org/10.1177/0261927X09351676>.
- [35] The National Academics of Sciences, Engineering, and Medicine 2018. *Data Science for Undergraduates: Opportunities and Options*. The National Academic Press.
- [36] Tran, X., Williams, J., Mitre, B., Walker, V. and Carter, K. 2017. Learning Styles, Motivation, and Career Choice: Insights for International Business Students From Linguistic Inquiry. *Journal of Teaching International Business*. 28, 3–4 (2017), 154–167. DOI:<https://doi.org/10.1080/08975930.2017.1384949>.

Admitting you have a problem is the first step: Modeling when and why students seek help in programming assignments.

Zhikai Gao
North Carolina State
University
zgao9@ncsu.edu

Collin Lynch
North Carolina State
University
cflynch@ncsu.edu

Bradley Erickson
North Carolina State
University
bericks@ncsu.edu

Sarah Heckman
North Carolina State
University
sarah_heckman@ncsu.edu

Yiqiao Xu
North Carolina State
University
yxu35@ncsu.edu

Tiffany Barnes
North Carolina State
University
tmbarnes@ncsu.edu

ABSTRACT

In computer science education timely help seeking during large programming projects is essential for student success. Help-seeking in typical courses happens in office hours and through online forums. In this research, we analyze students coding activities and help requests to understand the interaction between these activities. We collected student's help requests during coding assignments on two different platforms in a CS2 course, and categorized those requests into eight categories (including implementation, addressing test failures, general debugging, etc.). Then we analyzed the proportion of each type of requests and how they changed over time. We also collected student's coding status (including what part of the code changed and the frequency of commits) before they seek help to investigate if students share a similar code change behavior leading to certain type of help requests.

Keywords

Help Seeking, Categorization, Data Mining, CS2, Computer Science Education

1. INTRODUCTION

Help seeking is a complex cognitive skill involving metacognition and self-evaluation to identify the need for assistance, identification of problems for support, and formulating requests [1, 16, 15]. Help seeking is essential for learners to develop a better understanding of assignments or class content that they do not understand [15]. Effective help-seeking is a key part of learning and is associated with a capacity for self-regulated learning through monitoring and goal-setting among other aspects [22, 24]. Motivation in help-seeking has

Z. Gao, B. Erickson, Y. Xu, C. Lynch, S. Heckman, and T. Barnes. Admitting you have a problem is the first step: Modeling when and why students seek help in programming assignments. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 508–514, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853173>

been analyzed by a number of authors [19, 4, 25, 5, e.g] . These researchers have noted that learners who felt comfortable and skillful in relating to others were more likely to ask for help. Further studies of specific help-seeking behaviors (e.g. [17, 18, 21]) has included analyses of the interaction between help-seeking and instructor feedback. However, most of this prior research focused only on students' help seeking behaviours in isolation or in the context of relatively focused problem solving. It has not typically exemplified that help-seeking responds to prior problem-solving, and how these help requests affect their subsequent work.

In this study, our goal is to explore the motivations, help-seeking, and problem solving actions of computer science students seeking help for their programming projects. We chose to focus on coding for two reasons. First, computer science skills have become an increasingly important domain at all levels with increased demand at all grade levels and coding is an essential part of that [2]. Second, programming is a complex task that involves students in long-term complex problem solving which offers multiple opportunities for help-seeking and for a complex interrelationship between problem-solving, assistance, and outcomes. In contrast to prior research, we combined different types of student help seeking behaviours including attending office hour and posting on a public class forum. We analyze how students change their code when trying to receive help from others. Thus, in this work, we answer the following research questions.

- RQ1: What types of help are students seeking in online forums and office hour settings and how do they differ?
- RQ2: How does the frequency of help-seeking change over the course of students' complex assignments, and can we use the project stages or sub-goals to predict this help-seeking?
- RQ3: What types of coding behaviors do students engage in before seeking help?

In order to address these questions we began by identifying common help requests and coding behaviors. We manually

labeled help seeking requests from two different platform from a blended course. We then analyzed how these different types of help seeking requests changed during the coding project lifespan. Finally, we compared students' coding changes based on the commit frequency and the number of their passed test cases before and after they seek help.

2. RELATED WORK

2.1 Help Seeking Behaviours

As prior researchers have shown self-regulation of learning through modulation of affective, cognitive, and behavioral processes is an essential component of learning across domains [23]. This regulation and essential help-seeking is driven in part by students personal motivations [13, 14, 12], and by their general attitude toward learning [19]. In general students who have better performance in an educational environment tend to have better metacognitive skills and tend to seek help more frequently [10]. Ryan et al. [19], for example, investigated motivational influences on help seeking behavior in math classrooms, focusing on early teenagers' perception of the benefits and threats associated with such behaviour. They designed a survey for 203 seventh and eighth graders on perceptions of social and cognitive competence, achievement goals, attitudes, and avoidance of and adaptive help seeking behaviour. Their finding indicates social competence had an indirect effect on avoidance of help seeking. And the results illustrate the importance of linking cognitive and social characteristics of students to provide a full understanding of teenager help seeking.

Help seeking is an important factor to success in learning programming. Bumbacher et al. [3], for example, developed novel models to predict student learning gains based upon their semantic and structural features of their coding submissions. They found that these code features extracted from a single assignment can be used to predict whether or not students got help. Marwan et al. [11] focused on analyzing and classifying students' help-seeking behaviors. Based upon an analysis of student-system logfiles they proposed a taxonomy of unproductive help seeking behaviour in a programming environment. They then used these findings to design a hint interface that scaffolded appropriate help-seeking. Students using their platform were ultimately less than half as likely to produce unproductive help-seeking and thus improved overall.

2.2 Student Help Seeking Behaviour Analysis

Unlike in-person environments online help-seeking is far more open and produces a higher volume of data [9]. Prior studies of online learning have typically shown a positive relationship between help seeking behaviours and academic performance.

In-person or small-group online office hours are an important venue for support in traditional and blended courses. Guerrero et al. [8] noted that students fail to take advantage of office hours when they are available despite the fact that the use of office hours correlates with performance. Griffin et al. [7] extended this work by working to identify distinct factors that influence students' use of office hours. To that end they developed a survey with 625 valid responses from undergraduate students at a large public university. The results revealed that factors that significantly affect student

use of office hour vary with one exception: usefulness of instructional staff feedback. Thus, in this study, they suggest instructors provide more efficient feedback to solve students problems to encourage students to engage in office hour.

3. DATASET

3.1 Course Background

The data for this work originates from the Fall 2020 offering of a CS2 course at a research intensive university in the Southeast United States. Students in the course complete two projects, each worth 22% of their overall grade. Each project consists of two parts: 1) designing the system and creating a testing plan and 2) implementing the teaching staff (TS) UML diagram. Our focus will be on the second part of each project.

The course initially offered both online and in-person sections; however, due to the COVID-19 pandemic, all undergraduate courses covered in our dataset were moved online a few weeks into the semester. The overall structure of the course as well as the task deadlines remained the same with interactions including office hours moving to individual web meetings but retaining their overall structure.

As a first step in their course projects the students are required to develop UML diagrams for the problem task. However for the later coding stage the students must all follow the instructor-provided UML diagram. This allows them to fit a shared model for testing and evaluation. Students manage their repository using a Github¹ enterprise server. Whenever they push code, Jenkins², a continuous integration system, runs an Ant-driven build. Each build compiles the code, runs static analysis tools (Checkstyle³, PMD⁴, SpotBugs⁵), compiles the teaching staff test cases against the code to ensure it abides by the provided UML diagram, runs student-written tests to check for coverage metrics, runs teaching staff tests, and finally provides feedback to the student based on their status. For the purposes of our analysis we recorded the state of their code on each commit along with a record of the unit test results.

Over the course of the project, there are two intermediate milestones, or Process Points, for students to follow. Achieving the first two milestones provide a fraction of their project grade, while the final milestone defines the requirements to receive full credit.

- **Milestone 1 - Process Points 1:** students complete a compiling skeleton, at least one test case, and fully Javadoc (i.e. no Checkstyle notifications) their code.
- **Milestone 2 - Process Points 2:** students achieve 60% statement coverage on their self-written tests.
- **Milestone 3 - Done:** students achieve 80% statement coverage, have no static analysis notifications, and all tests are passing (both teaching staff and student written).

¹<https://github.com/>

²<https://www.jenkins.io/>

³<https://checkstyle.sourceforge.io/>

⁴<https://pmd.github.io/>

⁵<https://spotbugs.github.io/>

After achieving each of the milestones, the students receive feedback on their code based upon the shared tests. At the start, students only receive information about their compiling status and Checkstyle notifications related to developing Javadocs for their code. Once a student completes Milestone 1, they begin to see feedback about the remaining static analysis notifications. For completing Milestone 2, they begin seeing feedback regarding teaching staff test case failures. This feedback includes the number of tests passing or failing and for the failing tests it provides a hint to reproduce locally. For the purposes of our analyses we defined four project stages based upon these milestones. At the beginning of the project students are in stage 0, they then move to stage 1 after completing Milestone 1, and so on.

3.2 Help Seeking

Students in this course had two options for help, general questions could be posted to Piazza an online question and answer forum that was monitored by the instructors and took answers from other students; and office hours which were accessed via a help request system called My Digital Hand.

Piazza: At any time during the semester, students are able to post questions or comments on Piazza, an online forum for the class. When posting, students are required to select a category for their question. These categories correspond to logistics or each assignment, lab, or project. Students are able to make their posts private to instructors, but they are encouraged to post publicly so other students are able to answer questions or receive similar help. The teaching staff also uses Piazza as a place to post updates or announcements. We remove the teaching staff posts and filter posts down to only the second part of each project. We focus our attention only on the initial posting and remove all replies.

Office Hours During teaching staff office hours, students request help by filing a help ticket using My Digital Hand (MDH) an online support system [20]. Students seeking help through MDH fill out a form listing the tasks they are working on, what question they have or problem they are struggling with, and what steps they have taken thus far. The students are then placed in a queue which is monitored by the teaching staff. The teachers prioritize students to help based upon their questions and may help them individually or in small groups. After the interaction is complete the teachers will close the ticket and both they and the student can enter followup information to describe the advice given, rate the outcome, and any potential followups. Students who require more assistance may have their tickets re-opened or, more commonly, make a new ticket with additional questions. In our analysis, we first remove all tickets that were not related to the second part of the projects. Next, we remove any follow-up tickets or tickets that were re-opened with the same content.

3.3 Commit mining

After the semester concluded, we ran the BUILDDATACOLLECTOR⁶. This tool iterates over each commit from each repository and runs the Jenkins build. The output files are

⁶<https://github.com/SOS-CER/BuildDataCollector>

mined for relevant data. The data includes commit meta-data, static analysis notifications, information about all test cases, code counts, and coverage metrics. The data are stored in a SQL database.

4. METHODOLOGY

4.1 RQ1: Categorization of Help Requests

In developing our analysis we held the initial expectation that students' coding behaviors would differ based upon the type of help that they required. Thus, we began by classifying the students' help requests based upon their question content. In general students begin the project by seeking to understand the overall functionality of the system, and to build the general structure. They typically turn to authoring their own test cases based upon the functional goals with the goal of unlocking the instructor test cases on which their grade is partially based. Once these are unlocked students frequently use the test cases to drive their development process, as intended, so they often focus their questions around those tests. Across all of these stages they also face challenges with basic implementation errors and general static analysis notifications. Our goal was to separate students who were seeking to support with basic development tasks (e.g. debugging) or with specific test cases from those who were seeking to address deeper comprehension questions, or general notifications.

We therefore classified student help requests as follows:

- **General debugging and addressing issues:** students indicate they are receiving an error (i.e. null pointer exception) or describe unexpected behavior in their code.
- **Implementation and understanding:** students ask about how to implement some portion of the project or ask for clarification.
- **Improving test coverage:** students ask about how to improve their code coverage to achieve the 60% or 80% threshold.
- **Addressing TS test failures:** students indicate they are failing specific teaching staff test cases.
- **Addressing student-written test failures:** students indicate they are failing specific tests they wrote.
- **Addressing general test failures:** students indicate they are struggling with testing, but do not specify which type of test.
- **Addressing static analysis notifications:** students ask about Spotbugs, Checkstyle, or PMD notifications they receive.
- **Others or unclear:** students ask about unrelated topic (i.e. coding environment setup, documentation) or they are unclear with the type of help they need (i.e. a method name without explanation).

Manual Annotation To support our analysis we engaged three experienced TAs to manually tag the student questions across the two help contexts. We began by developing a shared coding process using a subset of Piazza and MDH posts. An

initial round of grading yielded a Fleiss' Kappa agreement of 0.73 for MDH data and 0.69 for Piazza data. After one round of iterative evaluation and agreement we achieved a final agreement of 0.81 for MDH data and 0.72 for Piazza. The primary area of disagreement in the Piazza data lies between questions in the Implementation and General Debugging categories. We therefore opted to perform additional segmentation based upon whether or not students showed evidence of code execution.

4.2 RQ2: Timeline analysis

After the categorization was complete, we examined the frequency of the students' help requests by type and the change in those frequencies over time. Our hypotheses in this analysis were:

- Hypothesis 1: At the early stages of the project (before the Milestone 2 deadline), students mainly ask Implementation questions.
- Hypothesis 2: After achieving Milestone 2, students mainly asked about General Debugging and TS Test Failure questions.

We believe that, consistent with instructor guidance, the students will try to understand the requirements of the project and how to implement each function first. Moreover, since Milestone 2 requires students to achieve 60% test coverage, it indicates that students need to finish most of the implementation to develop corresponding tests and reach Milestone 2. Therefore, we believe that prior to the Milestone 2 deadline, the most frequently asked questions should be Implementation related.

Moreover, after Milestone 2, students unlocked the TS test cases and also needed to develop more self-written tests to reach 80% code coverage. We believe that during this process, the students' main goal will be to correct their implementation based on the feedback of error messages and testing failures. Therefore, later in the projects, questions related to General Debugging and TS Test Failures should be more prevalent.

4.3 RQ3: Pre-help code analysis

In order to analyze the precursors of students' help requests we analyzed the state of the students' code on the last commit before they post a question to Piazza or file a ticket on MDH. We define these as the pre-help commit states. In analyzing these precursors we began by analyzing how frequently students made commits within 1 hour before their pre-help commit. We believe that a higher volume of pre-help commits indicate that the students are unlikely to be working on higher level implementation and are more focused on small-scale debugging or test passing.

In addition to the commit frequency, we also examined where code changes were made prior to help requests. Prior to requests for implementation help we imagine that most commits include changes to the core functionality, while help requests for coverage focus on changes to the student-written unit tests. This leads to the following hypotheses:

Category	MDH	Piazza	Overall
Implementation	334	281	615
General Debugging	392	316	708
TS test failures	127	307	434
Self-written test failures	30	90	120
General test failures	103	64	167
Improving test coverage	21	31	52
static analysis notifications	9	38	47
Others/Unclear	314	75	389
Total	1330	1202	2532

Table 1: Categorization Results

- Hypothesis 3: Before an Implementation request, student commit frequency is low and before a TS test failure request, the frequency is high.
- Hypothesis 4: Before an Implementation request, students are making changes to their source code and before a Coverage request, students are making changes to their test code.

5. RESULTS

5.1 RQ1

Table 1 lists the results of our final request categorization. In both of the help platform, the most popular type of request is General Debugging; We identified 392 (29.4%) MDH requests and 316 (26.3%) Piazza requests in this category. For office hours request, the next popular category is Implementation (25.1%), while for Piazza, there are more requests on Addressing TS Test Failures (25.5%) than Implementation (23.3%). The remaining categories are relatively uncommon with a frequency lower than 10% on each platform, except for the Others/Unclear category. We found a large amount of MDH requests containing very vague descriptions, which were categorized as Other/Unclear (23.6%). This is caused by the nature of the office hours process, the description we collected from MDH does not ask student to give very detailed information and students do not rely on it for getting the actual help; they would prefer to briefly describe their problem on MDH and elaborate on the detail orally when meeting with the teaching staff. This also matches with Gao's founding on the usage of MDH [6]. In piazza, since students more reliant on the description to get help, the amount of requests categorized as Others/Unclear is only 75 (6.2%).

5.2 RQ2

5.2.1 MDH Requests Timeline Analysis

Figure 1 shows how the student's help request types changed over time during office hours. Prior to the Milestone 1 deadline, students mainly asked the Implementation and General Debugging questions; during the next stage, before the Milestone 2 deadline, Implementation questions are dominating the help requests and maintain a very high number every single day. Then, after the Milestone 2 deadline, the amount of Implementation questions suddenly drop to less than 10 each day; while we witness a great increase in both the General Debugging and TS Test Failures, especially the General Debugging.

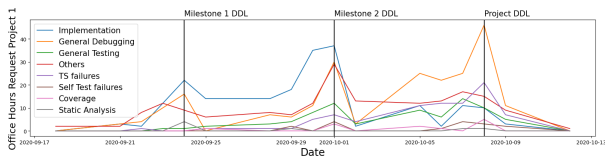


Figure 1: Project 1 Office Hours request timeline for each type.

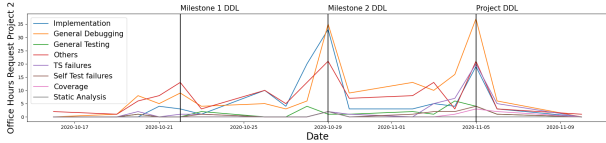


Figure 2: Project 2 Office Hours request timeline for each type.

Similarly, Figure 2 provides the project 2 office hour requests. This also contains a high number of Implementation questions leading up to the Milestone 2 deadline, while after the deadline, Implementation instantly becomes less common. However before Milestone 1, the General Debugging is also one of the dominating categories, with roughly the same amount as Implementation. After Milestone 2, General Debugging stays as the most common request while the amount of TS Test Failure requests increase similar to the first project.

Furthermore, both project 1 and project 2 have the most requests at the exact date of each deadline. The project deadlines have more requests than the Milestone 2 deadlines, which have more requests than the Milestone 1 deadlines. This observation shows that most students are completing their work on the deadline date and require more help.

5.2.2 Piazza Requests Timeline Analysis

Figures 3 and 4 show the number of help requests student made on Piazza each day during the two projects. The three vertical lines represent the milestones and final deadline dates.

Figure 3 shows the number of requests in Piazza that students made during project 1. We observed that the General Debugging category reaches the peak at all deadlines; the Implementation category is prominent during the first few stages; and the TS Test Failure category increases after the first milestone deadline and reaches its peak during the final deadline.

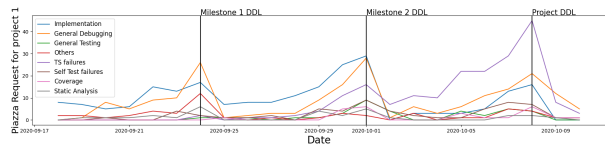


Figure 3: Project 1 Piazza request timeline for each type.

Figure 4 shows the number of requests in Piazza that students made during project 2. We observed that the General Debugging category always peaks at each deadline; the Gen-

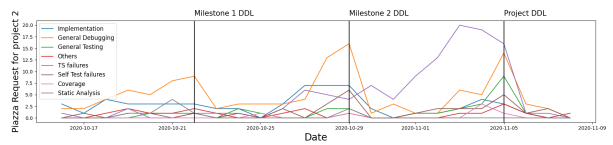


Figure 4: Project 2 Piazza request timeline for each type.

	Implementation	Coverage	Overall
src change	79.65%	60.78%	73.25%
test change	64.25%	84.31%	70.12%

Table 2: Percentage of pre-help commits with source or test code changes

eral Testing and Self-Testing categories peak at the project deadline; the TS Test Failure category raises rapidly from the Milestone 2 deadline to Project deadline; and the Implementation category increase before each deadline, but overall decreases after each subsequent deadline.

5.3 RQ3

5.3.1 Frequency of pre-help commits

When comparing the frequency of commits, we find that students who ask about TS Test Failures are on average making more commits in the last hour than students making other help-seeking requests, 4.6 commits and 2.9 commits respectively (p -value < 0.0001). Similarly, we see that students who seek help on Implementation requests are on average making fewer commits in the last hour than students who make any other request, 2.4 and 3.4 commits per hour respectively (p -value < 0.0001).

5.3.2 Code changes of pre-help commits

For the Implementation pre-help commits, 490 (79.65%) of those commits contains source code changes. Our chi-square test proves that Implementation requests have a significant higher number of pre-help commits with source code change (p -value < 0.05). For the Coverage pre-help commits, we found 43 (84.31%) of those commits contains test code changes. Similarly, we proved that this high number of commits with test code changes is significant than the rest of commits (p -value < 0.01).

6. DISCUSSION AND CONCLUSIONS

Our goal in this work was to investigate what kinds of requests students make when they seek help and what precursors exist in their code state before such requests are made.

In addressing RQ1, we categorized the help requests based on student's purpose. Our results shows that students are mostly asking questions about General Debugging, Implementation, and Addressing TS Test Failures. For office hours, General Debugging is the most popular category followed by Implementation followed by Addressing TS Test Failures. For Piazza requests, General Debugging are also the most common type, but Addressing TS Test Failures are more common than Implementation.

In RQ2, we analyzed the amount of help request each day and observed how it changed during different stages of the

project. We proved both of our hypothesis. Before Milestone 2, Implementation is the most common request (Hypothesis 1). After Milestone 2, General debugging is the most common request. We also see an obvious increase of TS Test Failure requests after Milestone 2 (Hypothesis 2).

In RQ3, we examined students' commits immediately prior to each help-request and define it as the pre-help commit. We firstly looked at how frequently students are making commits 1 hour before seeking help. Our analysis shows that the frequency of committing before an Implementation request is significantly lower and for TS Test Failures, it's significantly higher. Then, we analyzed where the students change their code right before the help request. We find that most students change their source code before Implementation requests and change their test code before Coverage-related requests.

Our current results shows student's coding behavior is quite predictable and it does change when they need different types of help. Instructor can use our conclusion as a start, and think about how to lead students to an efficient path when they encounter different types of difficulty throughout the projects.

7. LIMITATIONS AND FUTURE WORK

Our study only analyzed the data from a single CS2 course in a single offering. Additional analysis of future semesters and other courses would enhance our understanding of how stable these behaviors are across cohorts and projects. Additionally the quality of our analysis is limited by our post annotations. The high rate of disagreement for the Others/Unclear category was driven in part by how little information students included in their MDH requests. In future semesters we plan to address this problem by refining our analysis and addressing potential changes to the MDH platform to enforce more complex comments.

Additionally, our examination of the pre-help commits was focused on both general frequencies and gestalt features. In future studies we will conduct a more detailed analysis of the code status and align the contents of the help requests to specific code features and file locations. Finding such link of code features and request content would improve our understanding of when students seek help and potentially lead to an automotive model for detecting and predicting students who need help. Such a model would help instructors to actively and efficiently intervene student's behavior, and support a much effective help management for the course.

Besides the pre-help commits, we also plan to analyze changes that the students make to their code during and after help requests to get a better assessment of how they process and implement guidance. This work has long-term potential to guide automated feedback and to assist in general triage for help responses. For example, after we figure out an accurate way to measure the success of commits, we can combine it with this work and evaluate the effectiveness of each help-seeking interaction by monitoring when student make an successful commit after the help. The instructor can use this information to quickly identify which students need further guidance or analyze why the help is ineffective.

8. ACKNOWLEDGEMENTS

This material is based upon work supported by NSF under grants #1821475 "Concert: Coordinating Educational Interactions for Student Engagement" Collin F. Lynch, Tiffany Barnes, and Sarah Heckman (Co-PIs).

References

- [1] Vincent Aleven et al. "Toward meta-cognitive tutoring: A model of help seeking with a Cognitive Tutor". In: *International Journal of Artificial Intelligence in Education* 16.2 (2006), pp. 101–128.
- [2] Theresa Beaubouef and John Mason. "Why the high attrition rate for computer science students: some thoughts and observations". In: *ACM SIGCSE Bulletin* 37.2 (2005), pp. 103–106.
- [3] Engin Bumbacher et al. "Student coding styles as predictors of help-seeking behavior". In: *International Conference on Artificial Intelligence in Education*. Springer, 2013, pp. 856–859.
- [4] Yuk Fai Cheong, Frank Pajares, and Paul S Oberman. "Motivation and academic help-seeking in high school computer science". In: *Computer Science Education* 14.1 (2004), pp. 3–19.
- [5] Zhikai Gao, Sarah Heckman, and Collin Lynch. "Who Uses Office Hours? A Comparison of In-Person and Virtual Office Hours Utilization". In: *Proceedings of the 53rd ACM Technical Symposium V.1 on Computer Science Education*. SIGCSE 2022. Providence, RI, USA: Association for Computing Machinery, 2022, pp. 300–306. ISBN: 9781450390705. DOI: 10.1145/3478431.3499334. URL: <https://doi.org/10.1145/3478431.3499334>.
- [6] Zhikai Gao et al. "Automatically Classifying Student Help Requests: A Multi-Year Analysis." In: *International Educational Data Mining Society* (2021).
- [7] Whitney Griffin et al. "Starting the conversation: An exploratory study of factors that influence student office hour use". In: *College Teaching* 62.3 (2014), pp. 94–99.
- [8] Mario Guerrero and Alisa Beth Rod. "Engaging in office hours: A study of student-faculty interaction and academic performance". In: *Journal of Political Science Education* 9.4 (2013), pp. 403–416.
- [9] Stuart A Karabenick. "Classroom and technology-supported help seeking: The need for converging research paradigms". In: *Learning and instruction* 21.2 (2011), pp. 290–296.
- [10] Liisa Karlsson et al. "From novice to expert: Information seeking processes of university students and researchers". In: *Procedia-Social and Behavioral Sciences* 45 (2012), pp. 577–587.
- [11] Samiha Marwan, Anay Dombe, and Thomas W Price. "Unproductive help-seeking in programming: What it is and how to address it". In: *Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education*. 2020, pp. 54–60.
- [12] Richard S Newman. "Adaptive help seeking: A strategy of self-regulated learning." In: (1994).

- [13] Richard S Newman. “Children’s help-seeking in the classroom: The role of motivational factors and attitudes.” In: *Journal of educational psychology* 82.1 (1990), p. 71.
- [14] Richard S Newman. “Goals and self-regulated learning: What motivates children to seek academic help?”. In: *Advances in motivation and achievement* 7 (1991), pp. 151–183.
- [15] Richard S Newman. “The motivational role of adaptive help seeking in self-regulated learning”. In: *Motivation and self-regulated learning: Theory, research, and applications* (2008), pp. 315–337.
- [16] Richard S Newman and Mahna T Schwager. “Students’ help seeking during problem solving: Effects of grade, goal, and prior achievement”. In: *American Educational Research Journal* 32.2 (1995), pp. 352–376.
- [17] Ido Roll et al. “Improving students’ help-seeking skills using metacognitive feedback in an intelligent tutoring system”. In: *Learning and instruction* 21.2 (2011), pp. 267–280.
- [18] Ido Roll et al. “The help tutor: Does metacognitive feedback improve students’ help-seeking actions, skills and learning?” In: *International conference on intelligent tutoring systems*. Springer, 2006, pp. 360–369.
- [19] Allison M Ryan and Paul R Pintrich. “Should I ask for help?” The role of motivation and attitudes in adolescents’ help seeking in math class.” In: *Journal of educational psychology* 89.2 (1997), p. 329.
- [20] Aaron J. Smith et al. “My Digital Hand: A Tool for Scaling Up One-to-One Peer Teaching in Support of Computer Science Learning”. In: *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*. SIGCSE ’17. Seattle, Washington, USA: Association for Computing Machinery, 2017, pp. 549–554. ISBN: 9781450346986. DOI: 10.1145/3017680.3017800. URL: <https://doi-org.prox.lib.ncsu.edu/10.1145/3017680.3017800>.
- [21] Bram E Vaessen, Frans J Prins, and Johan Jeuring. “University students’ achievement goals and help-seeking strategies in an intelligent tutoring system”. In: *Computers & Education* 72 (2014), pp. 196–208.
- [22] Barry J Zimmerman. “Attaining self-regulation: A social cognitive perspective”. In: *Handbook of self-regulation*. Elsevier, 2000, pp. 13–39.
- [23] Barry J Zimmerman. “Dimensions of academic self-regulation: A conceptual framework for education”. In: *Self-regulation of learning and performance: Issues and educational applications* 1 (1994), pp. 33–21.
- [24] Barry J Zimmerman. “Models of self-regulated learning and academic achievement”. In: *Self-regulated learning and academic achievement*. Springer, 1989, pp. 1–25.
- [25] Akane Zusho et al. “Contextual determinants of motivation and help seeking in the college classroom”. In: *The scholarship of teaching and learning in higher education: An evidence-based perspective* (2007), pp. 611–659.

Going beyond “Good Job”: Analyzing Helpful Feedback from the Student’s Perspective.

M Parvez Rashid, Yunkai Xiao, Edward F. Gehringer
Department of Computer Science
North Carolina State University
Raleigh, NC, USA
{mrashid4, yxiao28, efg}@ncsu.edu

ABSTRACT

Peer assessment can be a more effective pedagogical method when reviewers provide quality feedback. But what makes feedback helpful to reviewees? Other studies have identified quality feedback as focusing on detecting problems, providing suggestions, or pointing out where changes need to be made. However, it is important to seek students’ perspectives on what makes a review helpful to a reviewee. This study explores the helpfulness of feedback from students’ perspectives when the feedback contained suggestions or mentioned problems or both. We applied natural language processing techniques to identify suggestions and problems mentioned in peer reviews. We also analyzed important text features that are associated with suggestions or problems detected by the peer feedback. The result showed that students are likely to find a review helpful if a suggestion is provided along with the problem mentioned in the feedback rather than simply identifying the problem.

Keywords

Peer assessment, neural-network, natural language processing, correlation coefficient, suggestions

1. INTRODUCTION

Peer assessment has been proven to be an effective learning approach in both face-to-face and distance learning classes. It is especially useful in massive open online courses (MOOCs) where the potentially overwhelming number of students has no fixed bound. All of these students must be assessed by someone, and there are only a limited number of staff. Peer assessment can be as accurate instructor assessment, since artifacts are reviewed by multiple assessors who can invest more time than a teacher could [17]. It can also provide timely feedback [1] that helps students to focus on their weaknesses. Peer assessors pick up some of the feedback workload for instructors, who can then offer more help to students who are in need.

M. P. Rashid, Y. Xiao, and E. F. Gehringer. Going beyond “good job”: Analyzing helpful feedback from the student’s perspective. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 515–521, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853179>

Peer assessors provide assessment in two forms. One is textual feedback, which takes the form of prose feedback to a peer. This is usually used as formative feedback. Another is numerical scores, on a Likert scale, which allows a summative grade to be calculated. Most online peer-assessment environments use both kinds of feedback. Studies have shown that students learn more from giving feedback than receiving it [13, 2, 7, 5] and giving feedback engage students in active learning [16]. It forces students to think metacognitively [4], and learn in-depth, as reviewing a peer requires a good hold on the topic [8].

However, the learning experience in a peer-review environment depends on the quality of reviews provided by the student peers. The goals of fairness and equity require that, insofar as possible, all students receive helpful formative feedback on their work. But, not all assessors provide constructive feedback, due to lack of knowledge in the topic or simply carelessness. To encourage and guide students in reviewing the artifacts, instructors typically need to scrutinize reviews manually. This consumes a good portion of the time that would be saved by having students provide quality feedback. An automated analysis could save considerable time.

A few studies [20, 19] have tried to lessen the instructors’ assessment burden by automatically detecting characteristics of a quality review. That raises the question of what defines a quality review. According to Nelson and Schunn [10] high-quality feedback consists of (i) identifying a problem and (ii) suggesting a solution. However, their finding was based on students’ performance and not from their (students’) perspective. It is important to identify whether “quality feedback” is actually helpful to the reviewees, based on students’ opinion of what feedback is helpful.

In this paper, we propose a method using natural language processing (NLP) and neural networks to automate the process of analyzing and classifying reviews to discover whether they contain suggestions and/or problems. We analyzed the words that are used to include suggestions or problems in feedback. Our goal is to answer the following research questions:

- **RQ1:** Can we build a model to accurately detect comments containing suggestions or detecting problems?
- **RQ2:** Are “quality comments”—those containing suggestions, detecting problems, or both—actually helpful from the student’s point of view?

- **RQ3: Can an automated process effectively identify helpful feedback?**

2. RELATED WORK

This section discusses related work on identifying the properties of a “quality“ or “helpful“ peer assessment.

Nelson and Schunn [10] examined five features of feedback (summarization, specificity, explanations, scope, and affective language) that constitute good-quality reviews, and the correlations among them. Their study divided the features of feedback into cognitive and affective components. According to their findings, summarization, specificity, explanations, and scope are cognitive in nature. Cognitive features of a review are expected to most strongly affect understanding. This explanation helped us to identify suggestions and problem detection as a property of quality feedback.

An approach to improve review quality is to provide the reviewer with a rubric defining the characteristics of a quality review. Jaco du Toit [3] conducted a study to identify the impact of peer review on essay assignments. The study showed that giving students a rubric describing the characteristics of a good essay can provide them with the insight to produce better quality assessments than they would otherwise produce. But this study did not specify the qualities of a good review. When they received poor reviews, they were confused about the quality of their work, sometimes feeling a false sense of accomplishment. Rashid et al [15] analyzed rubric items to determine which of them induce peer reviewers to write quality feedback using NLP approaches. In their work quality feedback was identified if the review text contained a suggestion, detected a problem problem, or was localized (pinpointing the place where a revision should be made).

McGrath and Taylor did a study on students’ perception of helpful feedback for writing performance [9]. Their study defined quality feedback (“developed feedback”) as clear, specific, and explanatory in nature. They measured students’ perception of developed feedback by having them rate the feedback on a Likert scale. The results showed that students rated developed feedback highly for helpfulness.

A survey of 44 students done by Weaver showed that, in order to use the feedback, students needed advice (suggestions)[18]. The analysis of the feedback content and students’ responses uncovered that vague feedback (e.g., “Good job”) is unhelpful, lacking in guidance (void of suggestions), or focused on the negative (mentioning only problems), or was unrelated to assessment criteria.

Ramachandran et al. [14], developed an automated system to evaluate reviews and show how they compared to other reviews for the same assignment. They extracted attributes like relevance to the submission, content, coverage, tone, and volume of feedback to identify a good-quality review. They constructed word-order graphs to compare the reviews with submission text and extract features from the reviews.

To identify localization and make suggestions to improve the review Nguyen et al. [11] applied natural language processing techniques. They provided real-time formative feedback

to reviewers on how to localize their review comments.

Zingle et al. [20] used neural-network approaches to find suggestions in the review text, and compared them against rule-based NLP approaches. In a similar work Xiao et al. [19] used NLP techniques with several ML and neural-network approaches to identify problem statements in review text. Our work takes this a step further and asks whether it is enough for a review to detect problems, or whether reviews that also make suggestions are more helpful.

3. DATA

Machine-learning and neural network-based models can perform as well or as badly as the data they are given. However, obtaining good labeled data is expensive. For the purpose of our experiment, we have collected labeled datasets for comments with three different characteristics:

- detects a problem: A review comment is labeled yes or no according to whether it detects a problem.
- contains a suggestion: A review comment is labeled yes or no according to whether it contains a suggestion.
- is helpful: A review comment is labeled yes or no depending on whether the reviewee found it helpful.

We acquired this labeled peer-review data from the Expertiza system in a systematic manner. Expertiza is a system to support different kinds of communications that are involved in the peer-assessment process. It supports double-blind communications between authors and reviewers, assessment of teammate contributions, and evaluations by course staff.

For the purpose of this study, we collected the data from Object-Oriented Design and Development course at NC State University for about three years. This course used the Expertiza system to manage the peer-review assessment process for evaluating the students. In each semester, this course typically assigns three peer-reviewed assignments to students, who work in teams consisting of two to four members. Even though the assignments are done in a group setting, the submissions are reviewed by individual students from other groups. After receiving the reviews from peers, teams revise their work and resubmit it for grading. The second round of the assessment is generally summative, where along with textual comments, the peer-reviewers assign scores to the submission.

Generally, a small number of people cannot annotate a large dataset. It is better to have a large number of people each undertake a small number of annotation tasks; this lessens the chance that an annotator will become fatigued and assign inaccurate labels. We engaged students in the labeling task by offering a small amount of extra credit. After receiving peer feedback, students were asked to label the feedback to identify whether the reviewer mentioned a problem or suggested a solution. They were also asked whether they considered the feedback to be helpful. In different assignments, students were asked to label the feedback for different characteristics; the same comments were not necessarily labeled for all three characteristics. After labeling was complete, the course instructor and TAs spot-checked the data

Table 1: Sample review comment and annotations done by students ('1' indicates 'yes' and '0' indicates 'no')

Review Comment	Detects Problem
The Travis CI Build is Failing as of now. No conflicts as per the GitHub report.	1
Yes, the explanation is elaborative and complete.	0
Since the build failed, I would not recommend adding it to the production server yet.	1

Review Comment	Gives Suggestion
Test Plan is too verbose. Trivial areas can be trimmed off.	1
The team needs to look into Travis CI log & 1	1
Many test cases in terms of controllers, but none for models.	0

Review Comment	Is Helpful
The build is failing due to 4 failures in the model specs.	1
The writeup is clear.	0
Since the build failed, I would not recommend adding it to the production server yet.	1

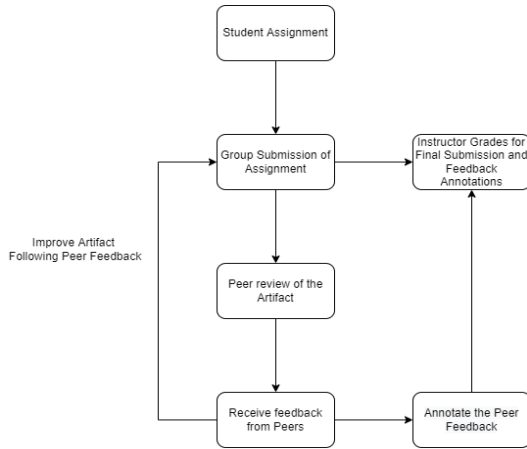


Figure 1: Flow diagram of peer review and feedback annotation process

that each student labeled. If any labels were found to be incorrect, the data labeled by that student was excluded from the dataset.

Since the reviews were done on team projects, and labeling was done individually, two to four students had the opportunity to label (or “tag”) the same review comments. If multiple students did tag the same comment, inter-rater reliability (IRR) could be calculated. We chose Krippendorff’s α [6] as the metric for IRR. We chose this metric because it is not impacted by missing ratings, which were common since not all students availed themselves of the extra-credit opportunity. In an effort to use only the most reliable labeling, we included only labels that were assigned (or not assigned) by all the students in the team that was reviewed. This allowed us to raise Krippendorff’s α of our dataset from 0.696 to 1. Figure 1 shows the peer-review and annotation process.

Following the described process we accumulated 18,392 annotations for problem detection, 7,416 for suggestion-detection and 3,970 for helpfulness-detection datasets. All the three datasets have an equal ratio of the binary class labels (i.e., they are balanced). Sample comments from the three datasets are shown in Table 1

4. METHOD

Our goal in this study is to analyze students’ perspectives on helpful comments that mentioned problems and/or suggestions. To conduct the study, we had students annotate comments on the basis of whether they found them helpful. We need an automated process to identify those review comments that contain suggestions and/or problem statements. We first train a model (the problem-detection model) to classify reviews that contain a problem statement by training and testing with the problem-detection dataset. We build a second model (the suggestion-detection model) to classify the presence of suggestions in a review comment by using the suggestion-detection dataset. As model performance matters, we applied several ML and neural-network models to pick the most accurate models for annotating the helpfulness-detection dataset. Figure 2 shows the annotation process of the helpfulness-dataset using the models.

When approaching a classification problem by any type of machine-learning (ML) or neural network models, there are many different approaches to choose from. No one model is best for all problems. In our study, we have chosen Support Vector Machine (SVM), Random forest (RF), classical ML models and compared their performance with Bi Directional Long Short-term Memory (Bi-LSTM), and Bidirectional Encoder Representations from Transformers (BERT) models. We used TF-IDF for ML models and Global Vectors for Word Representation (GloVe) for Bi-LSTM to perform word vectorization. Before we applied any word vectorization techniques, we cleaned the text by removing URLs, stop words, and applying stemming. We use our problem-detection dataset and suggestion-detection dataset on these model with 80:10:10 ratio for training, testing and validation.

4.1 Classical Machine-Learning Models

4.1.1 Input Embedding with TF-IDF

Machine-learning models are suitable for capturing complex relationships between the input data. But they require numeric input. The review data that we have in our dataset is textual. We have to convert them to numbers and also allow the model to capture the important features of the text. One way to do that is term frequency-inverse document frequency (TF-IDF). TF-IDF measures the importance of a word in a document using statistical calculation. If a word appears more times in a document the importance of the word in the document increases proportionally. We used

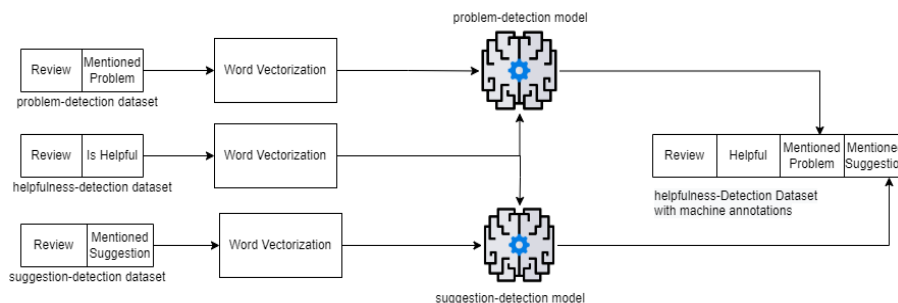


Figure 2: Annotation process of the helpfulness-dataset for mentioned problem and suggestions in the comments using models. The models were trained for detecting problems or suggestions mentioned in the review text. The training datasets were annotated by human (students).

scikit-learn [12] library to implement TF-IDF and vectorize the words in the feedback.

4.1.2 SVM

SVM is very popular for high accuracy and low computational cost. For a classification problem between two classes, SVM maximizes the margin of the separation plane between the two classes. We provided the feature vector of the reviews converted by TF-IDF to the SVM model to classify the review for having a particular property (contains problem or suggestion in the comment). We applied a grid search to find the best inverse regularization parameter C .

4.1.3 RF

We used Random Forest for its popularity to make more accurate classification with a simple approach. RF makes an ensemble decision from a forest consisting of multiple uncorrelated decision trees. The general idea of the RF is that the decision from individual decision trees increase the accuracy of overall result. We varied the number of decision trees and depth of the trees to get the best result. We used TF-IDF for making feature vectors from the review text.

4.2 Neural-Network Models

4.2.1 Input Embedding

Neural network models are popular for text classification tasks. However, to improve the performance of the neural-network models on the text data, it is necessary to represent the data that is suitable for the model to work with, and without losing the underlying latent relations among the features of the data. For our experiment we have used GloVe with Bi-LSTM. GloVe not only measures the statistical significance of words, it also considers the statistical co-occurrence and semantic relation of the words.

4.2.2 Bi-LSTM

Bi-Long Short-Term Memory is in general used for sequential data classification tasks. It is a good fit for peer-review texts. Review comments are sequential data, and the words of the text have latent semantic and contextual relations with each other. As Bi-LSTM model takes input from both right and left direction of the text, it can capture the relationship between the words in texts occurring in any order.

Table 2: Hyperparameters of Models

Model	Hyperparameter
SVM	$c=1$
RF	tree = 100 max depth = 4
Bi-LSTM	maximum text length = 300 Embedding = 300d Hidden layer activation = ReLu dropout = 0.4 optimizer = Adam Output layer activation = Sigmoid Epoch=20
BERT	optimizer = AdamW Learning rate = $2e-5$ Epoch=4

4.2.3 BERT

BERT is based on Transformer model and use attention mechanism to learn the contextual relations of the words in a sentence. Being a bi-directional input reader, BERT learns the context of word in sentence by considering words occurring before and after.

5. RESULTS

In this study, if a feedback comment mentions problems and/or suggestions, we are considering it to be quality feedback. Our first step is to construct two separate models where one identifies whether feedback contains a problem statement and another identifies whether feedback contains a suggestion. To identify the best-performing models we trained and tested the performance of several classical ML models and neural-network models and compared their performance.

RQ1: Can we build a model to accurately detect comments containing suggestions or detecting problems?

Figure 3 reports the comparison of the F1-score values of the classical machine-learning (ML) models and neural-network models on the problem-detection dataset and suggestion-detection dataset. To compare the performance of the models we use the F1-score, as this represents the harmonic mean of precision and recall.

- **On the problem-detection dataset:** Among the classical

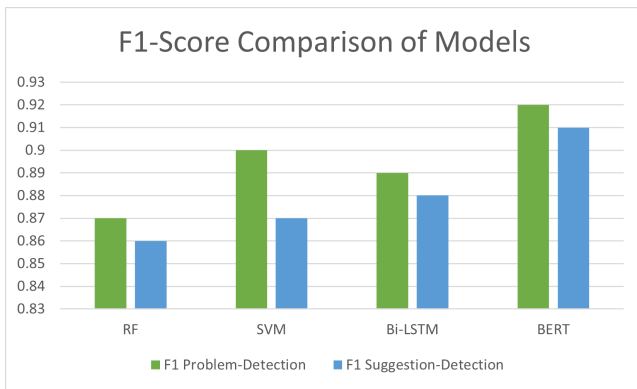


Figure 3: F1-score comparison to measure performance on classifying review text on problem detection and suggestion detection using classical ML and neural-network models. In overall F1-score comparison, the BERT model shows the best performance.

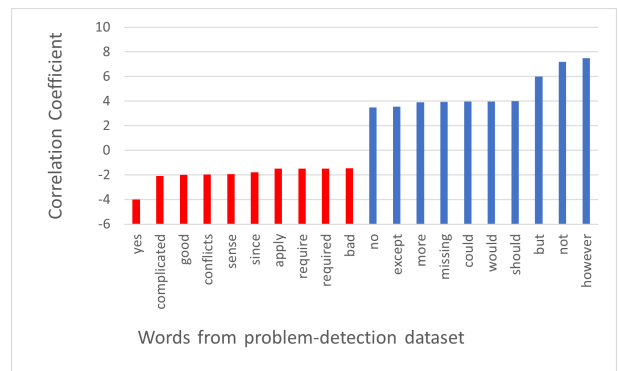
ML models SVM made the highest f1-score 0.90 and among the neural-network models; BERT obtains the overall highest F1-score, 0.92.

- **On the suggestion-detection dataset:** BERT achieved the highest F1-score, 0.91. Among the classical ML models, SVM achieved the highest F1-score, 0.87.

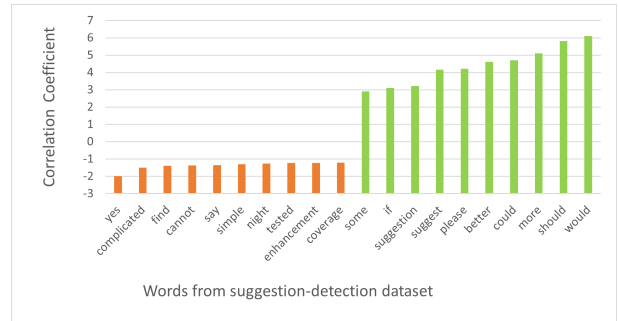
To gain a deeper insight into the words that are highly correlated with text where a problem mention or suggestion is mentioned, we analyzed the top 10 positive and negative correlation coefficient values calculated by the SVM model. Figure 4(a) shows the coefficient values that the problem-detection model has calculated for various words. Note that it has positive coefficient values for words such as “however”, “but”, and “not”. In the English language these words are more likely to be used when stating problems. Similarly words like “yes”, “completed” and “good” are not likely to occur in a problem statement. Figure 4(b) shows that the suggestion-detection model has positive coefficient values for words like “should”, “would”, “more”, “suggest”. These words are likely to be used in suggestions. On the other hand, words “yes”, “completed”, and “cannot” are more likely not to be used to express suggestions; thus they have negative coefficient values.

As BERT outperformed all other models on both problem and suggestion datasets, we trained two separate BERT models to annotate the feedback comments contained in the helpfulness-detection dataset. The BERT-created annotations recorded whether each comment in the helpfulness-detection dataset detected a problem or offered a suggestion. The models annotated each comment with either “1” or “0”, indicating having the property or not. We perform an **and**-operation using the BERT-created annotations. If both the problem and suggestion were mentioned in a comment the **and**-operation yields 1 otherwise 0. The resulting helpfulness-detection dataset is shown in Table 3.

RQ2: Are “quality comments”—those containing suggestions, detecting problems, or both—actually helpful from the stu-

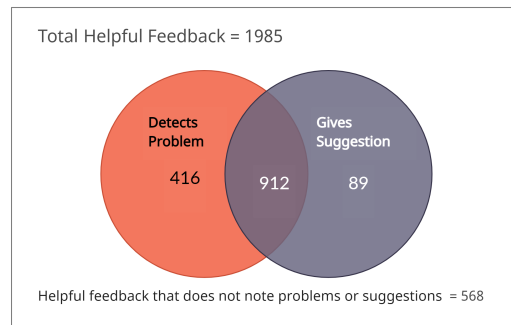


(a)

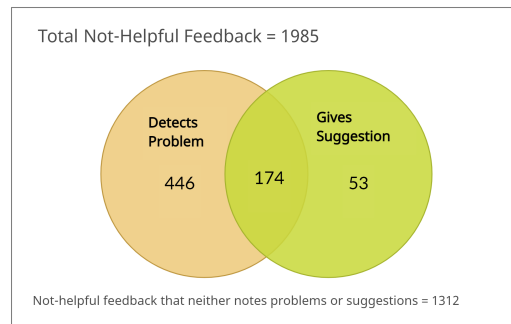


(b)

Figure 4: Top 10 positive and negative coefficient value of words from the problem-detection and suggestion-detection datasets



(c)



(d)

Figure 5: Venn diagram of helpful feedback annotated for mentioned suggestion and/or problem

Table 3: Table shows sample comments from helpfulness-detection dataset and corresponding annotations. Note that “Is Helpful” annotations are done by humans (students), while “Detects Problem” and “Gives Suggestion” are annotated by the BERT model. “Contains Problem and Suggestion” is from **anding** the “Detects Problem” and “Gives Suggestion” columns.

Review Comment	Is Helpful (human-annotated)	Detects Problem (machine-annotated)	Gives Suggestion (machine-annotated)	Contains Problem and Suggestion (and-operation)
The build is failing due to 4 failures in the model specs.	1	1	0	0
The writeup is clear.	0	0	0	0
Since the build failed, I would not recommend adding it to the production server yet.	1	1	1	1
I would recommend adding more code for helping following their changes.	1	0	1	0

dent’s point of view?

After we computed the annotations for problem detection and suggestions, we did a Venn diagram analysis on the updated helpfulness-detection dataset. The diagrams illustrate the overlap of comments that both detect a problem and offer a suggestion. Figure 5(c) shows that 1,985 comments in the helpfulness-detection dataset were annotated by students as being helpful. Among the helpful comments, 1,417 were annotated for having problems and/or suggestions mentioned in the feedback. Out of these 1,417 helpful comments, 912 of them were machine-annotated as containing both problem detection and suggestions. A total of 568 helpful comments did not have any problem or suggestion mentioned, based on machine-annotation.

On the other hand, out of the 1,985 comments that were human-annotated as not helpful [Figure 5(d)], 673 comments were annotated as having either a problem and/or suggestion mentioned. Among those 673 comments, only 174 were annotated as having both suggestion and problem mentioned. A total of 1,312 comments that did not have any problem or suggestion mentioned were annotated as not helpful by the students.

To summarize the Venn diagram analysis, comments that the students found helpful mostly detected problems and/or contained suggestions. However, among those comments noting suggestions and/or problems, students annotated as helpful mostly comments that *both* pointed out problems and gave suggestions. This indicates that peer feedback is more helpful to the students when a suggestion is given in a comment that detects a problem. On the flip side, Figure 5(b) suggests that students rarely find comments helpful when they do not mention any problem or contain a suggestion.

RQ3: Can an automated process effectively identify helpful feedback?

A key question is whether comments automatically annotated as “quality” (meaning that they both identified a problem and gave a suggestion) were the same comments that the students considered helpful (that they manually labeled as helpful). Among the comments that the students considered helpful, 64% of them both mentioned a problem and gave a suggestion. Conversely, of the comments that the students labeled as not helpful, 66% of them neither mentioned a problem nor contained a suggestion.

The results indicate that the automated annotation per-

formed by the BERT model can be very effective in predicting which comments students will consider helpful. While it can’t deliver an actual count of helpful comments in a particular review, that is not important. It *can* determine whether the feedback provided by the reviewer contains a substantial number of quality comments. That is what is needed to automatically detect helpful reviews.

6. CONCLUSION

This study constitutes the first analysis of the helpfulness of peer-assessment feedback from student’s perspective. Feedback that mentions problems or includes suggested changes was considered to be quality feedback. We used natural language processing (NLP) techniques in conjunction with several ML and neural networks to identify quality peer feedback. We systematically collected and scrutinized 18,392 comments mentioning problems, 7,416 comments containing suggestions, and 3,970 comments that were annotated by humans (students) as being helpful.

Using the annotated dataset, we trained our ML and neural-network models to identify quality feedback. For identifying suggestions and problems mentioned in the review text, the BERT model outperformed the other models. As the BERT model focuses on the important features of the text, it was best at identifying suggestions and problems in the feedback. We used the BERT model to automatically annotate comments as mentioning problems or making suggestions, and compared these annotations with comments that students had manually annotated as being helpful. We also analyzed important words that are frequently present in comments mentioned a problem or suggestion.

A key finding of this study is that the students find review comments more helpful when peer reviewers both mention problems in the reviewed artifact and provide suggestions on how to resolve the issue. We can use a state-of-the-art BERT model to automatically identify the helpful review comments.

It should not be hit-or-miss whether students receive helpful reviews on their submitted work from peer reviewers. This study helps identify helpful feedback and therefore, help students to improve their work.

7. REFERENCES

- [1] J. Cambre, S. Klemmer, and C. Kulkarni. Juxtapeer: Comparative peer review yields higher quality feedback and promotes deeper reflection. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pages 1–13, 2018.
- [2] Y. D. Çevik. Assessor or assessee? investigating the differential effects of online peer assessment roles in the development of students’ problem-solving skills. *Computers in Human Behavior*, 52:250–258, 2015.
- [3] J. du Toit. Enhancing the quality of essays through a student peer-review process. In *International Conference on Innovative Technologies and Learning*, pages 459–468. Springer, 2019.
- [4] E. F. Gehringer. A survey of methods for improving review quality. In *International Conference on Web-Based Learning*, pages 92–97. Springer, 2014.
- [5] M. H. Graner. Revision workshops: An alternative to peer editing groups. *The English Journal*, 76(3):40–45, 1987.
- [6] K. Krippendorff. *Content analysis: An introduction to its methodology*. Sage publications, 2018.
- [7] L. Li and V. Grion. The power of giving feedback and receiving feedback in peer assessment. *All Ireland Journal of Higher Education*, 11(2), 2019.
- [8] K. Lundstrom and W. Baker. To give is better than to receive: The benefits of peer review to the reviewer’s own writing. *Journal of second language writing*, 18(1):30–43, 2009.
- [9] A. L. McGrath, A. Taylor, and T. A. Pychyl. Writing helpful feedback: The influence of feedback type on students’ perceptions and writing performance. *Canadian Journal for the Scholarship of Teaching and Learning*, 2(2):5, 2011.
- [10] M. M. Nelson and C. D. Schunn. The nature of feedback: How different types of peer feedback affect writing performance. *Instructional Science*, 37(4):375–401, 2009.
- [11] H. Nguyen, W. Xiong, and D. Litman. Iterative design and classroom evaluation of automated formative feedback for improving peer feedback localization. *International Journal of Artificial Intelligence in Education*, 27(3):582–622, 2017.
- [12] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.
- [13] R. Rada et al. Collaborative hypermedia in a classroom setting. *Journal of Educational Multimedia and Hypermedia*, 3(1):21–36, 1994.
- [14] L. Ramachandran, E. F. Gehringer, and R. K. Yadav. Automated assessment of the quality of peer reviews using natural language processing techniques. *International Journal of Artificial Intelligence in Education*, 27(3):534–581, 2017.
- [15] M. P. Rashid, E. F. Gehringer, M. Young, D. Doshi, Q. Jia, and Y. Xiao. Peer assessment rubric analyzer: An nlp approach to analyzing rubric items for better peer-review. In *2021 19th International Conference on Information Technology Based Higher Education and Training (ITHET)*, pages 1–9. IEEE, 2021.
- [16] K. Topping. Peer assessment between students in colleges and universities. *Review of educational Research*, 68(3):249–276, 1998.
- [17] K. J. Topping. Peer assessment. *Theory into practice*, 48(1):20–27, 2009.
- [18] M. R. Weaver. Do students value feedback? student perceptions of tutors’ written responses. *Assessment & Evaluation in Higher Education*, 31(3):379–394, 2006.
- [19] Y. Xiao, G. Zingle, Q. Jia, H. R. Shah, Y. Zhang, T. Li, M. Karovaliya, W. Zhao, Y. Song, J. Ji, et al. Detecting problem statements in peer assessments. In *Proceedings of The 13th International Conference on Educational Data Mining (EDM 2020)*, pages 704–709, 2020.
- [20] G. Zingle, B. Radhakrishnan, Y. Xiao, E. Gehringer, Z. Xiao, F. Pramudianto, G. Khurana, and A. Arnav. Detecting suggestions in peer assessments. *International Educational Data Mining Society*, 2019.

The AI Teacher Test: Measuring the Pedagogical Ability of Blender and GPT-3 in Educational Dialogues

Anaïs Tack
Stanford University
atack@cs.stanford.edu

Chris Piech
Stanford University
piech@cs.stanford.edu

ABSTRACT

How can we test whether state-of-the-art generative models, such as Blender and GPT-3, are good AI teachers, capable of replying to a student in an educational dialogue? Designing an AI teacher test is challenging: although evaluation methods are much-needed, there is no off-the-shelf solution to measuring pedagogical ability. This paper reports on a first attempt at an AI teacher test. We built a solution around the insight that you can run conversational agents in parallel to human teachers in real-world dialogues, simulate how different agents would respond to a student, and compare these counterpart responses in terms of three abilities: speak like a teacher, understand a student, help a student. Our method builds on the reliability of comparative judgments in education and uses a probabilistic model and Bayesian sampling to infer estimates of pedagogical ability. We find that, even though conversational agents (Blender in particular) perform well on conversational uptake, they are quantifiably worse than real teachers on several pedagogical dimensions, especially with regard to helpfulness (Blender: Δ ability = -0.75 ; GPT-3: Δ ability = -0.93).

Keywords

student-teacher dialogue, conversational agents, chatbots, Blender, GPT-3, evaluation methods, pairwise comparisons, Bayesian Bradley-Terry model

1. INTRODUCTION

Conversational agents (or chatbots) offer promising opportunities for education. They can fulfill various roles (such as intelligent tutors and service-oriented assistants) and pursue different objectives (e.g., improving student skills, boosting student motivation, and increasing instructional efficiency) [20]. Among all of these different vocations of an educational chatbot, the most prevalent one is the AI teacher helping a student with skill improvement and providing more opportunities to practice. Some recent meta-analyses have even reported a significant effect of chatbots on skill im-

provement, for example in language learning [1]. What is more, current advances in AI and natural language processing have led to the development of conversational agents that are founded on more powerful generative language models. Blender [17], for instance, is a state-of-the-art open-domain chatbot trained to blend skills such as being empathetic and knowledgeable [18], which are undeniably important characteristics of a good AI teacher. Furthermore, the current state-of-the-art in natural language generation is GPT-3 [4], a 175B-parameter model that is able to multitask different language generation skills (such as conversation). The astonishing power of GPT-3 is that it can perform these skills with few-shot in-context learning, merely from seeing a short prompt describing the task at hand (e.g., *The following is a conversation with an AI assistant.*). Emergent models such as GPT-3 have been described as *foundation models* since they serve as the “common basis from which many task-specific models are built via adaptation” [2, p.7].

Despite these promising opportunities, the use of powerful generative models as a foundation for downstream tasks also presents several crucial challenges. In the educational domain in particular, it is important to ascertain whether that foundation is solid or flimsy. Bommasani et al. [2, pp.67-72] stressed that, if we want to put these models into practice as AI teachers, it is imperative to determine whether they can (a) speak to students like a teacher, (b) understand students, and (c) help students improve their understanding. Consequently, there is a critical need to establish good evaluation methods of AI teachers. This is a hard problem because there is no off-the-shelf and universal solution to measuring teaching ability and effectiveness.

Therefore, we took on the challenge of designing an AI teacher test and conducted a pilot study. We ran Blender and GPT-3 in parallel to human teachers in language and mathematics educational dialogues, observed how they responded to a student, and compared these counterpart responses in terms of pedagogical ability. The major contributions of this work are as follows:

1. We pose the AI Teacher Test Challenge.
2. We implement a human-in-the-loop pairwise comparison test as a first attempt at an AI Teacher Test.
3. Our results show quantitatively how far conversational agents, particularly Blender and GPT-3, are behind human teachers in terms of pedagogical ability, despite them performing well on conversational uptake.

A. Tack and C. Piech. The AI teacher test: Measuring the pedagogical ability of blender and GPT-3 in educational dialogues. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 522–529, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853187>

Our solution has several strengths: (1) it leverages the proven reliability of comparative judgments in education [10, 12], (2) it incorporates a Bayesian sampling method that allows us to attribute an ability score to a teacher response, whilst ensuring normality and providing a belief in our estimates, and (3) it produces scores and ranks that could be used to develop autonomous methods. We open-source our work, code, and data.¹

2. THE AI TEACHER TEST CHALLENGE

Consider the following scenario, which is illustrated in Figure 1. Two agents, a student and a teacher, are interacting in an educational setting. The student is working to improve a specific skill (e.g., the use of phrasal verbs) in a given domain (e.g., English language). The teacher could be either a human agent or an artificial agent who is helping the student with improving this skill. The student and teacher each take turns, resulting in a sequence of student-teacher dialogic pairs. This student-teacher dialogue is open-ended: for a given student utterance, there exists a variety of ways in which we could imagine a teacher agent to respond. For example, Figure 1 shows three possible replies to a student utterance: the actual teacher’s response and two completions that were automatically generated from a state-of-the-art language model. It is clear to see that, in the space of possible replies, not all responses will be equally preferable. Some responses may be more characteristic of a teacher, some may be taking up more from the student’s utterance, and some may be more helpful. In this scenario, we are interested in the following challenge: given a space of possible responses (either human or artificially generated), evaluate a reply in terms of pedagogical ability and estimate this score relative to other replies.

2.1 Desiderata

We think that a good AI teacher test should at least account for the following aspects. Firstly, the test should be able to evaluate a teacher agent’s response in context. At minimum, the test should consider the preceding student utterance. Additionally, the test could also take into consideration the entire preceding dialogue and surrounding educational setting. Secondly, the test should be able to score the agent’s response with respect to several pedagogical abilities. Following Bommasani et al. [2, pp.67-72], we believe that the test should consider the following three abilities: whether the agent can speak like a teacher, understand the student, and help the student. Finally, the test should also be able to consider other possibilities (which may be better or worse) and rank the teacher’s response in comparison to these. In this way, the test could also be used to suggest one or more ways in which a response could be enhanced in terms of the three abilities listed above.

Unfortunately, standard methods of evaluating automatically generated language and conversational agents do not meet our desiderata. Perplexity, for example, measures how well a generative model is able to sample a given response from its probability distribution. However, it does not consider the preceding utterance (desideratum #1). Other metrics such as BLEU and F1 score measure the n-gram overlap between a generated response and a correct response. By

¹<https://github.com/anaistack/ai-teacher-test>

The following is a conversation between a student and a teacher working on a language exercise.

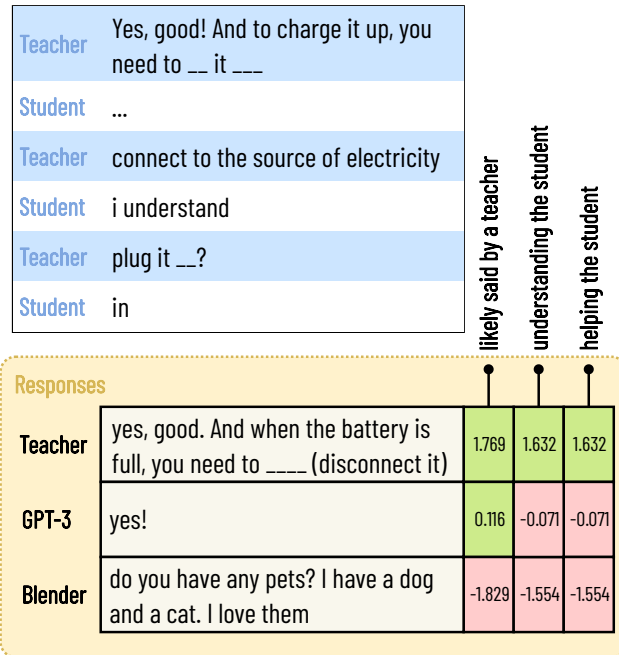


Figure 1: Illustration of the AI Teacher Test Challenge: Estimates of Pedagogical Ability and Rankings of Human and AI Teachers Replying to a Student in an Educational Dialogue

contrast, our test is open-ended (see above) and does not presuppose the existence of a correct response. Recently, Pillutla et al. [15] introduced MAUVE, an evaluation metric for open-ended language generation. Because this metric uses the Kullback–Leibler divergence, it cannot be used to compare two specific language utterances (desideratum #3). Most importantly, none of these methods meet our second desideratum, which is to score an agent’s response with respect to several pedagogical abilities.

2.2 Related Work

We can gain insight into measuring pedagogical ability from prior work into assessing human teachers. Educational research is abundant in methods for evaluating teacher effectiveness, ranging from teacher self-reports and interviews to classroom observations, student evaluation surveys, and tests of student achievement [9, 14]. However, not all of these methods seem easily applicable to assessing AI teachers. It is obvious that evaluating AI teacher effectiveness from self-reports and interviews would be a difficult thing to do. We could, however, resort to systematic observations of AI teachers, human evaluation surveys, and measures of student outcome.

Other studies have focused on the possibility of measuring ability in teacher language. Demzky et al. [6], for instance, examined several ways of determining how well a teacher replies to a student in student-teacher interactions. Their data comprised 2,246 student-teacher dialogic pairs taken from the National Center for Teacher Effectiveness Main

Study (NCTE)², a three-year long observation of mathematics instruction. First, they collected human evaluations of *conversational uptake*, a measure of how well the teacher’s reply expanded on the student’s utterance (e.g., by acknowledging, reformulating, elaborating), as illustrated below.

Student: Seven plus seven is fourteen.
Teacher: Okay, so you doubled. You did your doubles first. Okay. Fourteen plus eight?
(Uptake = high)

Besides human evaluations of uptake, Demszky et al. [6] also developed an automated method that could predict uptake as a next-utterance classification task. They fine-tuned a BERT language model [7] and found a significant correlation ($\rho = .54$) with human evaluations.

This automated measure of conversational uptake can serve as a solid baseline for our study. First, the next-utterance classification predicts uptake based on the preceding student utterance and, therefore, meets our first desideratum. Second, conversational uptake also somehow measures whether a speaker understands the interlocutor. If a teacher’s response strongly expands on the student’s utterance (i.e., high uptake), it can be deduced that the teacher was able to understand the student. As such, it measures one of the three pedagogical abilities targeted in our second desideratum. Finally, because we can run the predictive model on different responses to the same student utterance and compare these responses in terms of uptake, the measure meets our third and final desideratum.

3. OUR AI TEACHER TEST

As a possible solution to the AI teacher challenge described in Section 2, we adopted the following method. First, we ran Blender and GPT-3 on real-world educational dialogues and simulated responses to student utterances. We then paid human raters to compare pairs of responses on several different pedagogical dimensions. Finally, we ran a probabilistic model to compute aggregate scores. In addition, we also ran the model developed by Demszky et al. [6] on our data in order to compare our scores to predictions of uptake.

3.1 Student-Teacher Dialogues

Table 1: Datasets of Student-Teacher Interactions

Domain	Dataset	Dialogues	Dialogic Pairs
Language	TSCC [5]	102	4439
Mathematics	Uptake [6]	0	2246

The two datasets used in this study are listed in Table 1. The *Educational Uptake Dataset* compiled by Demszky et al. [6] includes 2,246 dialogic pairs sampled from the NCTE transcripts (see Section 2.2). The complete dialogue transcripts, however, have not yet been made available. The *Teacher-Student Chatroom Corpus (TSCC)* compiled by Caines et al. [5] includes 102 anonymized student-teacher dialogues in second language education. Each chatroom is a lesson where a teacher converses with a student in order to work on a language exercise and assess the student’s English

²<https://doi.org/10.3886/ICPSR36095.v3>

language proficiency. The corpus includes 13,215 turns and 130 turns on average per dialogue. Each utterance is annotated with several metadata, including conversational organization (e.g., opening, closing, eliciting, scaffolding, and revision) and teaching focus (e.g., vocabulary). Figure 1 shows an example excerpt of a teacher’s eliciting, scaffolding, and revision. It should be noted, however, that the TSCC dialogues include many consecutive utterances by either the student or the teacher. Therefore, the data were slightly adapted for this study: all successive utterances by the same speaker were concatenated into one turn such that each conversation was composed of alternating dialogic pairs. As a result, the data included 4,439 student-teacher pairs.

3.2 Simulating Agent Responses

For each dialogic pair in the student-teacher dialogues, we automatically generated AI teacher responses. We used the ParLAI framework [13] to load the student-teacher dialogues, to generate responses to each student utterance, and to compute several standard evaluation metrics. In this study, we focused on two models. We ran several Blender models (90M, 400M, 3B, 9B parameters) on the language (TSCC) and mathematics (Uptake) educational dialogues. We implemented a new agent that made requests to the OpenAI API in order to obtain generated responses for each student utterance. Each request included a mandatory prompt with instructions for GPT-3 (*The following is a conversation with a teacher. The teacher is polite, helpful, professional, on topic, and factually correct.*), the preceding dialogue history (restricted to meet the maximum number of tokens per request), and the student’s utterance. We obtained completions from the smallest (Ada) and largest (Davinci) models.

3.3 Measuring Pedagogical Ability

After collecting AI teacher responses in educational dialogues, we collected evaluations of pedagogical ability via an online survey. First, participants were given a short introduction and a consent form. Then, participants were given the same example to familiarize themselves with the task at hand. In the following comparative judgment task (Figure 2), 15 items were randomly and evenly selected from a pool of relevant items. Each item had three components: a dialogue context (limited to 100 tokens), one comparison of two teacher replies, and three questions targeting a pedagogical ability (speak like a teacher, understand the student, and help the student). For each participant, one pairwise comparison was randomly selected from three possible combinations (Teacher vs. Blender, Teacher vs. GPT-3, or Blender vs. GPT-3) and the order of the comparative pair was randomly shuffled.

Item Selection. A crucial challenge in the evaluation process was to pinpoint those teacher utterances that were important to evaluate. In the student-teacher dialogues described in Section 3.1, not all teacher utterances were necessarily relevant. In fact, many conversational turns were not pertaining to any educational goal, such as opening sequences, closing sequences, and other chit-chat. From the 6,685 eligible dialogic pairs, only those utterances were selected where the teacher was actually eliciting and scaffolding the student’s understanding. Additionally, short utterances that comprised of single words or sentence fragments

The following is a conversation between a student and a teacher working on a math problem.

Student: Can I take a calculator?

Imagine the teacher replies with either one of the following answers.

A	B
Teacher: Yes. You can take anything you need. Just make sure you bring it back. Go ahead. Pack it up.	Teacher: Of course you can. You can use it to calculate the amount of time you have left to live.

Which of these replies is **more likely said by a teacher**?

A B I cannot tell

In which reply is the teacher **understanding the student more**?

A B I cannot tell

In which reply is the teacher **helping the student more**?

A B I cannot tell

Figure 2: Screenshot of the Comparative Judgment Task

(e.g., *Perfect!, Yay!*) were also excluded.³ Furthermore, the results of a pilot study with eight evaluators highlighted that the dialogic pairs taken from the Uptake dataset were difficult to evaluate because there was no informative context. Consequently, we focused only on the TSCC dataset for the comparative judgment task, carefully screened the corpus for relevant items and informative dialogue contexts, and ended up with a sample of 52 items.

Participants. We recruited a sample of 120 participants from Prolific Academic, a crowdsourcing platform developed at Oxford University. Participants were prescreened to ensure a balanced gender representation (50% female, 50% male). Study participants were aged 19 to 66 ($M = 33$, $SD = 11.3$; female: $M = 32.4$, $SD = 10.9$; male: $M = 33.5$, $SD = 11.7$) and resided in the United Kingdom ($n = 86$) or the United States ($n = 34$). On average, participants had an excellent Prolific score of 99.2% ($SD = 1.4$; female: $M = 99.1$, $SD = 1.6$; male: $M = 99.3$, $SD = 1.3$) and took 18 minutes to complete the survey ($SD = 11.2$; female: $M = 18.9$, $SD = 11.1$; male: $M = 17.3$, $SD = 11.4$). Because the tasks required a fair amount of cognitive involvement (reading the dialogue, reading different replies, comparing different options), we estimated that the survey would take about 30 minutes. We then used the default payment rate of £7.50/h. Participants were paid according to estimated study completion time (£3.75 for 30 minutes).

Agreement. There was a high observed agreement between evaluators on the example given before the comparative judgment task (Figure 2). Most agreed that option A (the true teacher response) was *more likely said by a teacher* (95%), *understanding the student more* (83%), and *helping the student more* (86%).

³It should be noted that this exclusion criterion did not apply to the generated responses. As shown in Figure 1, some generated responses comprised of single words or sentence fragments (e.g., *yes!*). Although this could be seen as giving an advantage to the teacher responses, it was only meant to focus our test on more expressive teacher language. In a future study, we might try to capture the full range of teacher language, from single words to complex utterances.

Outlier Detection. To detect potential outliers among the evaluators, we identified those who consistently chose option A or B in the paired comparisons. This first-position (or “home-field” advantage) effect was detected by estimating an intercept parameter α_0 in the model described below. However, instead of estimating different α parameters for each teacher response (combining the scores of all evaluators), we reversed the method and computed different α parameters for each evaluator (combining the scores for all items evaluated by the evaluator). Evaluators were excluded when the credible interval around the intercept was above or below zero, which indicated that they were biased towards selecting either option A (CI above zero) or option B (CI below zero). Based on this outlier detection method, the data from seven evaluators were removed. The remaining data included 4,782 comparisons from 113 evaluators and 10.9 evaluations on average for each pair (Teacher vs. Blender, Teacher vs. GPT-3, or Blender vs. GPT-3).

Bayesian Bradley-Terry Model. A Bradley-Terry model [3] is a probabilistic model that predicts the outcome of one or more pairwise comparisons. Consider n items (i.e., a student utterance and preceding dialogue), a set of t possible responses (i.e., Teacher, Blender, GPT-3) to each item, and a set of m abilities (i.e., speak like a teacher, understand the student, help the student). For each item $l \in [n]$ and for each ability $k \in [m]$, we inferred a latent parameter α_{ikl} for each possible teacher response $i \in [t]$. The outcome y_{ijkl} was an independent Bernoulli variable with a parameter $p_{ijkl} \in [0, 1]$ measuring the chance that, for an item l and an ability k , teacher response i would be preferred over teacher response j , for all $i, j \in [t]$ and $i \neq j$. This probability was defined as

$$p_{ijkl} := \sigma(\alpha_{ikl} - \alpha_{jkl}) \Rightarrow \log \frac{p_{ijkl}}{1 - p_{ijkl}} = \alpha_{ikl} - \alpha_{jkl} \quad (1)$$

where σ is the logistic function $\sigma(x) = \frac{1}{1 + e^{-x}}$ and α_i, α_j are the latent parameters that measure the strengths of i and j respectively. In case of ties (the *I cannot tell* option), the outcome was picked uniformly at random. We used an extended version of the basic Bradley-Terry model including an intercept parameter $\alpha_0 \in \mathbb{R}$, which measures a “home-field” advantage.

$$p_{ijkl} := \sigma(\alpha_{0kl} + \alpha_{ikl} - \alpha_{jkl}) \quad (2)$$

If $\alpha_0 > 0$, there was a greater chance that the evaluator would pick the first element in the comparison. If $\alpha_0 = 0$, there was no order effect. To infer the latent parameters $\vec{\alpha}_{kl} = (\alpha_{0kl}, \dots, \alpha_{tkl})$, we adopted a Bayesian approach by drawing samples from the posterior $p(\alpha|y) \propto p(y|\alpha)p(\alpha)$ with a non-conjugate prior distribution, $\alpha \sim \mathcal{N}(0, 1)$. We used Stan [19, 16] to compute posterior means and 95% HDI (Highest Density Interval) credible intervals from 4,000 simulations using Hamiltonian Monte Carlo (HMC) sampling [8] and the NUTS (No-U-Turn Sampler) algorithm [11]. For each simulation, the estimated ability parameters were used to rank each response on each item and for each ability.

4. RESULTS

4.1 Baseline: Conversational Uptake

We start our analyses with a comparison of conversational uptake in human and AI teacher responses, for the two student-teacher dialogue datasets presented in Section 3.1.

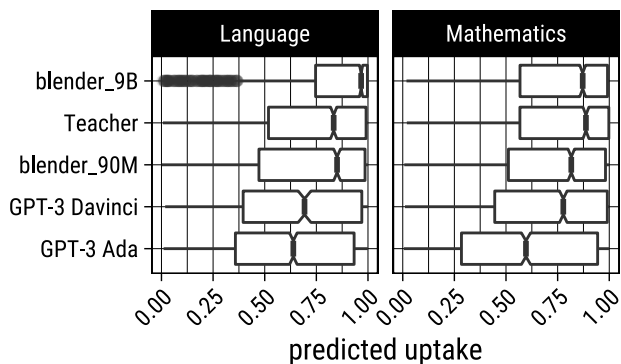


Figure 3: Predicted Uptake of Human and AI Teacher Responses in Language and Math Educational Dialogues

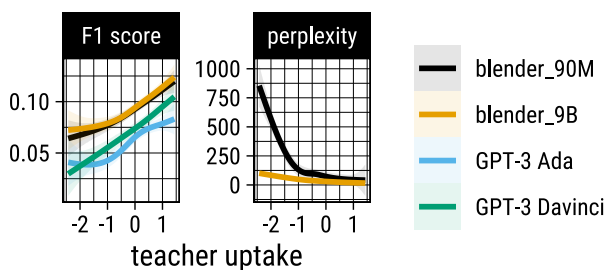


Figure 4: Associations Between Generative Performance (Model Perplexity, F1 Unigram Overlap) and True Teacher Uptake (Z-Score) in Mathematics Educational Dialogues

Figure 3 shows the predicted uptake for the smallest and largest Blender and GPT-3 models, compared to the actual teacher’s responses. The results show that the largest Blender model (with 9B parameters) outperformed all others for both the language (TSCC) and mathematics (Uptake) educational dialogues. This suggests that Blender tended to generate better next utterances to student utterances.

Figure 4 zooms in on the AI teacher responses in the mathematics educational dialogues. Several correlation analyses were run to examine the association between generative performance (perplexity and F1 score) and the human annotations of teacher uptake collected by Demszky et al. [6]. Perplexity (lower is better) indicates how well the model can generate a linguistic utterance from its probability distribution, whereas F1 score (higher is better) indicates the unigram overlap between the generated response and the teacher’s response. There was a negative, statistically significant, and large correlation between model perplexity and true teacher uptake, as measured by Pearson’s product-moment correlation coefficient, $r = -0.31$, 95% CI [-0.34, -0.26], $t(1996) = -14.32$, $p < .001$. Similarly, there was a positive, statistically significant, and small correlation between F1 unigram overlap and true teacher uptake, $r = 0.16$, 95% CI [0.12, 0.20], $t(1996) = 7.35$, $p < .001$. In other words, Blender tended to generate better responses in cases where the actual teacher was also judged to have given a better response (more uptake). Moreover, this association

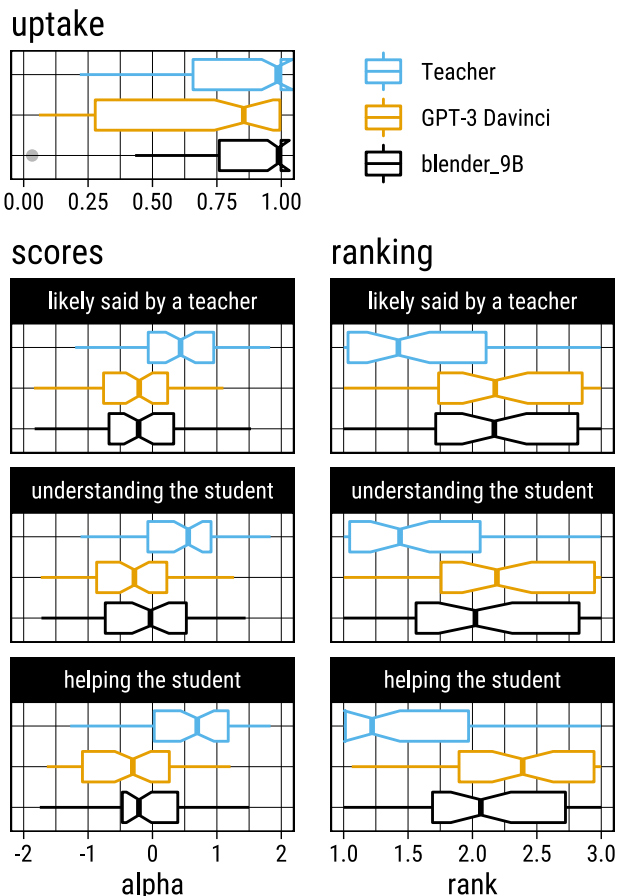


Figure 5: Bayesian Estimates and Rankings of Pedagogical Ability in Replying to a Student in an Educational Dialogue, Compared to Predictions of Conversational Uptake

between generative performance and teacher uptake was observed for all Blender and GPT-3 models (see Figure 4). These findings suggest that some student utterances may be simply easier to reply to, for both human and AI teachers.

4.2 Our Test: Pedagogical Ability

We now focus all following analyses on the selection of teacher responses that were compared in terms of pedagogical ability. Figure 5 shows a boxplot of the expected values of α (and associated rankings) for each possible response to a student utterance on the three pedagogical dimensions. The figure also compares these scores to predictions of conversational uptake. In terms of conversational uptake, the results showed no significant differences between human and AI teachers, as indicated by the overlapping notches in the boxplot. In terms of pedagogical ability, however, a one-way ANOVA revealed a statistically significant difference between human teachers and AI teachers on the three dimensions cited above, $F(2, 144) = 13.1$, $p < .001$, $F(2, 144) = 11.8$, $p < .001$, $F(2, 144) = 22.3$, $p < .001$, respectively.⁴ Tukey’s HSD post hoc test for multiple comparisons showed that, compared to the actual teacher, the mean

⁴A Shapiro-Wilk test showed that the assumption of normality was not violated for any of the three pedagogical

Table 2: Pearson Correlations Between Uptake and Ability

	r	t	df	p
likely said by a teacher	.35	3.47	85	<.001
understanding the student	.38	3.82	85	<.001
helping the student	.33	3.27	85	.002

Table 3: Percentage of Replies with a Positive Ability or where the 95% CI Excludes Zero (Either Above or Below)

Agent	Ability	$\alpha > 0$	$0 \notin \text{CI}$
Teacher	speak like a teacher	69%	8%
Teacher	understand the student	71%	6%
Teacher	help the student	78%	14%
Blender 9B	speak like a teacher	41%	6%
Blender 9B	understand the student	45%	10%
Blender 9B	help the student	35%	8%
GPT-3 Dav.	speak like a teacher	35%	6%
GPT-3 Dav.	understand the student	35%	2%
GPT-3 Dav.	help the student	33%	12%

ability of Blender was significantly lower for speaking like a teacher ($\Delta \alpha = -0.60$, [95CI $-0.93, -0.26$], $p < .001$), understanding the student ($\Delta \alpha = -0.55$, [95CI $-0.90, -0.20$], $p < .001$), and helping the student ($\Delta \alpha = -0.75$, [95CI $-1.10, -0.40$], $p < .001$). Similarly, compared to the actual teacher, the mean ability of GPT-3 was significantly lower for speaking like a teacher ($\Delta \alpha = -0.67$, [95CI $-1.00, -0.33$], $p < .001$), understanding the student ($\Delta \alpha = -0.67$, [95CI $-1.02, -0.32$], $p < .001$), and helping the student ($\Delta \alpha = -0.93$, [95CI $-1.28, -0.58$], $p < .001$). As for Blender and GPT-3, there was no statistically significant difference between the two when it came to speaking like a teacher ($\Delta \alpha = 0.071$, $p = .41$). Overall, Blender seemed better at understanding ($\Delta \alpha = +0.12$) and helping ($\Delta \alpha = +0.18$) the student but these differences were not significant ($p = .47$, $p = .53$, respectively).

Table 2 shows that our estimates of pedagogical ability were significantly correlated with conversational uptake. Interestingly, the highest correlation was observed for the ability of understanding the student. This result was not surprising: as previously noted in Section 2.2, uptake also somehow measures whether a speaker understands the interlocutor.

Finally, we examined the pedagogical ability of human and AI teachers on a per-item basis. Table 3 shows the percentage of teacher replies for which the expected value of α was positive. For 69-78% of dialogue contexts, the actual teacher’s response was positively evaluated and, therefore, highly ranked. For helpfulness in particular, 78% of human teacher responses were positively evaluated. There was also a higher percentage (14%) of certainty in the estimates of helpfulness. Nevertheless, there were many cases (22-31% of dialogue contexts) where the actual teacher’s response was negatively evaluated as well as cases (33-45%) where artificially generated responses were positively evaluated. The following example illustrates a case where the response generated by Blender was largely preferred over the

dimensions, $W = 0.99$, $p = 0.76$, $W = 0.99$, $p = 0.19$, $W = 0.99$, $p = 0.15$, respectively.

actual teacher’s response.

Teacher:	People are looking forward to it = there is a lot of ... ?
Student:	excitement
#1 Blender:	Excitement is a good one! People are very excited about it. Do you have any more? - likely said by a teacher = 1.52 - understanding the student = 1.32 - helping the student = 1.52
#2 Teacher:	ok good! Coronavirus has caused a lot of ... among the population. - likely said by a teacher = -0.90 - understanding the student = -0.57 - helping the student = -0.61
#3 GPT-3:	excitement about the next World Cup... - likely said by a teacher = -0.61 - understanding the student = -0.60 - helping the student = -0.90

5. CONCLUDING DISCUSSION

How well are state-of-the-art conversational agents, such as Blender and GPT-3, capable of replying to a student in an educational dialogue? When it comes to uptaking from and expanding on a student’s utterance, Blender comes out on top, outperforming the actual teacher and GPT-3. Based on the results of our AI teacher test, we come to similar conclusions. Although our test does not corroborate that Blender can actually outperform a human teacher, there is nevertheless a closer gap with human performance when it comes to understanding the student. Blender scores noticeably better on this specific pedagogical dimension, with a higher percentage of positively evaluated responses. These findings may be attributed to Blender’s particular training objective, namely blended skill talk. By learning to be more empathetic, Blender might be incidentally learning to take up more from and be more understanding of its interlocutor. By contrast, the results of our AI teacher test show that GPT-3 performs quantifiably worse than Blender and significantly worse than real teachers on all measured abilities, despite its proven capacity for few-shot in-context learning. What is more, both Blender and GPT-3 are well behind human performance when it comes to helping the student.

A secondary finding of our AI teacher test is that not all human teacher responses are necessarily positively evaluated. Even though the AI teacher responses generally fall short regarding pedagogical ability, we could still leverage generated responses as a means of sampling and recommending potentially better responses.

The solution proposed in this paper is surely not a perfect test, but it is a first step towards building much-needed evaluation methods.

6. ACKNOWLEDGMENTS

This research was supported by a fellowship of the Belgian American Educational Foundation (to the first author) and by a grant from Stanford HAI. We thank Andrew Caines, Dora Demszky, Noah Goodman, and our colleagues for their valuable help and suggestions. We thank all anonymous reviewers for their insights that improved the paper.

References

- [1] S. Bibauw, W. Van den Noortgate, T. François, and P. Desmet. Dialogue systems for language learning: A meta-analysis. *Language Learning & Technology*, 26(1): accepted, 2022.
- [2] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill, E. Brynjolfsson, S. Buch, D. Card, R. Castellon, N. Chatterji, A. Chen, K. Creel, J. Q. Davis, D. Demszky, C. Donahue, M. Doumbouya, E. Durmus, S. Ermon, J. Etchemendy, K. Ethayarajh, L. Fei-Fei, C. Finn, T. Gale, L. Gillespie, K. Goel, N. Goodman, S. Grossman, N. Guha, T. Hashimoto, P. Henderson, J. Hewitt, D. E. Ho, J. Hong, K. Hsu, J. Huang, T. Icard, S. Jain, D. Jurafsky, P. Kalluri, S. Karamcheti, G. Keeling, F. Khani, O. Khattab, P. W. Kohd, M. Krass, R. Krishna, R. Kuditipudi, A. Kumar, F. Ladhak, M. Lee, T. Lee, J. Leskovec, I. Levent, X. L. Li, X. Li, T. Ma, A. Malik, C. D. Manning, S. Mirchandani, E. Mitchell, Z. Munyikwa, S. Nair, A. Narayan, D. Narayanan, B. Newman, A. Nie, J. C. Niebles, H. Nilforoshan, J. Nyarko, G. Ogut, L. Orr, I. Papadimitriou, J. S. Park, C. Piech, E. Portelance, C. Potts, A. Raghunathan, R. Reich, H. Ren, F. Rong, Y. Roohani, C. Ruiz, J. Ryan, C. Ré, D. Sadigh, S. Sagawa, K. Santhanam, A. Shih, K. Srinivasan, A. Tamkin, R. Taori, A. W. Thomas, F. Tramèr, R. E. Wang, W. Wang, B. Wu, J. Wu, Y. Wu, S. M. Xie, M. Yasunaga, J. You, M. Zaharia, M. Zhang, T. Zhang, X. Zhang, Y. Zhang, L. Zheng, K. Zhou, and P. Liang. On the Opportunities and Risks of Foundation Models. Technical report, Stanford University, Center for Research on Foundation Models (CRFM), Aug. 2021.
- [3] R. A. Bradley and M. E. Terry. Rank Analysis of Incomplete Block Designs: I. The Method of Paired Comparisons. *Biometrika*, 39(3/4):324, Dec. 1952. ISSN 00063444. doi: 10.2307/2334029.
- [4] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020.
- [5] A. Caines, H. Yannakoudakis, H. Edmondson, H. Allen, P. Pérez-Paredes, B. Byrne, and P. Buttery. The teacher-student chatroom corpus. In *Proceedings of the 9th Workshop on NLP for Computer Assisted Language Learning*, pages 10–20, Gothenburg, Sweden, Nov. 2020. LiU Electronic Press.
- [6] D. Demszky, J. Liu, Z. Mancenido, J. Cohen, H. Hill, D. Jurafsky, and T. Hashimoto. Measuring Conversational Uptake: A Case Study on Student-Teacher Interactions. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1638–1653, Online, 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.130.
- [7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423.
- [8] S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth. Hybrid Monte Carlo. *Physics Letters B*, 195(2):216–222, Sept. 1987. ISSN 0370-2693. doi: 10.1016/0370-2693(87)91197-X.
- [9] L. Goe, C. Bell, and O. Little. *Approaches to Evaluating Teacher Effectiveness: A Research Synthesis*. National Comprehensive Center for Teacher Quality, June 2008.
- [10] S. Heldsinger and S. Humphry. Using the Method of Pairwise Comparison to Obtain Reliable Teacher Assessments. *Australian Educational Researcher*, 37(2): 1–19, Aug. 2010. ISSN 0311-6999.
- [11] M. D. Homan and A. Gelman. The No-U-turn sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo. *The Journal of Machine Learning Research*, 15 (1):1593–1623, Jan. 2014. ISSN 1532-4435.
- [12] M. Lesterhuis, S. Verhavert, L. Coertjens, V. Donche, and S. De Maeyer. Comparative judgement as a promising alternative to score competences. In E. Cano, G. Ion, and J. Keengwe, editors, *Innovative Practices for Higher Education Assessment and Measurement: Advances in Higher Education and Professional Development*. IGI Global, 2017. ISBN 978-1-5225-0531-0 978-1-5225-0532-7. doi: 10.4018/978-1-5225-0531-0.
- [13] A. Miller, W. Feng, D. Batra, A. Bordes, A. Fisch, J. Lu, D. Parikh, and J. Weston. ParlAI: A dialog research software platform. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 79–84, Copenhagen, Denmark, Sept. 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-2014.
- [14] D. Muijs. Measuring teacher effectiveness: Some methodological reflections. *Educational Research and Evaluation*, 12(1):53–74, Feb. 2006. ISSN 1380-3611, 1744-4187. doi: 10.1080/13803610500392236.
- [15] K. Pillutla, S. Swayamdipta, R. Zellers, J. Thickstun, S. Welleck, Y. Choi, and Z. Harchaoui. MAUVE: Measuring the Gap Between Neural Text and Human Text using Divergence Frontiers. In *Advances in Neural Information Processing Systems 34 Pre-Proceedings (NeurIPS 2021)*, pages 1–35, Nov. 2021.
- [16] A. Riddell, A. Hartikainen, and M. Carter. PyStan (3.3.0). PyPI, Mar. 2021.

- [17] S. Roller, E. Dinan, N. Goyal, D. Ju, M. Williamson, Y. Liu, J. Xu, M. Ott, K. Shuster, E. M. Smith, Y.-L. Boureau, and J. Weston. Recipes for building an open-domain chatbot. 2020. doi: 10.48550/ARXIV.2004.13637.
- [18] E. M. Smith, M. Williamson, K. Shuster, J. Weston, and Y.-L. Boureau. Can you put it all together: Evaluating conversational agents' ability to blend skills. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2021–2030, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.183.
- [19] Stan Development Team. Stan Modeling Language Users Guide and Reference Manual (v2.29.0), 2022.
- [20] S. Wollny, J. Schneider, D. Di Mitri, J. Weidlich, M. Rittberger, and H. Drachler. Are We There Yet? - A Systematic Literature Review on Chatbots in Education. *Frontiers in Artificial Intelligence*, 4:654924, July 2021. ISSN 2624-8212. doi: 10.3389/frai.2021.654924.

Automatic Classification of Learning Objectives Based on Bloom's Taxonomy

Yuheng Li
Centre for Learning Analytics
Monash University, Australia
yuheng.li@monash.edu

Mladen Raković*
Centre for Learning Analytics
Monash University, Australia
mladen.rakovic@monash.edu

Boon Xin Poh
Monash University, Australia
pohjessica96@hotmail.com

Dragan Gašević
Centre for Learning Analytics
Monash University, Australia
dragan.gasevic@monash.edu

Guanliang Chen*
Centre for Learning Analytics
Monash University, Australia
guanliang.chen@monash.edu

ABSTRACT

Learning objectives, especially those well defined by applying Bloom's taxonomy for Cognitive Objectives, have been widely recognized as important in various teaching and learning practices. However, many educators have difficulties developing learning objectives appropriate to the levels in Bloom's taxonomy, as they need to consider the progression of learners' skills with learning content as well as dependencies between different learning objectives. To remedy this challenge, we aimed to apply state-of-the-art computational techniques to automate the classification of learning objectives based on Bloom's taxonomy. Specifically, we collected 21,380 learning objectives from 5,558 different courses at an Australian university and manually labeled them according to the six cognitive levels of Bloom's taxonomy. Based on the labeled dataset, we applied five conventional machine learning approaches (i.e., naive Bayes, logistic regression, support vector machine, random forest, and XGBoost) and one deep learning approach based on pre-trained language model BERT to construct classifiers to automatically determine a learning objective's cognitive levels. In particular, we adopted and compared two methods in constructing the classifiers, i.e., constructing multiple binary classifiers (one for each cognitive level in Bloom's taxonomy) and constructing only one multi-class multi-label classifier to simultaneously identify all the corresponding cognitive levels. Through extensive evaluations, we demonstrated that: (i) BERT-based classifiers outperformed the others in all cognitive levels (Cohen's κ up to 0.93 and F1 score up to 0.95); (ii) three machine learning models – support vector machine, random forest, and XGBoost — delivered performance comparable to the BERT-based classifiers; and (iii) most of the binary BERT-based classifiers (5 out of 6) slightly outperformed the multi-class multi-label BERT-

based classifier, suggesting that separating the characterization of different cognitive levels seemed to be a better choice than building only one model to identify all cognitive levels at one time.

Keywords

Learning Objectives, Bloom's Taxonomy, Classification, Machine Learning, BERT

1. INTRODUCTION

A learning objective is a clear and specific statement defining knowledge and skills that learners are expected to acquire after completing an educational activity [19]. A well-articulated learning objective can benefit course designers, instructors and learners. For instance, learning objectives can inform course design, as they often signal how course materials should be organized to ensure a suitable sequencing of instruction and optimize learning activities throughout the semester. Instructors can utilize learning objectives to assess learners' progress; meanwhile, learners can use learning objectives to get an overview of knowledge and skills they should possess after receiving instruction [38], and to support their studying for an exam, e.g., by developing questions for self-testing prior to an exam [2].

Educators in many courses create learning objectives that reflect knowledge/skills of different levels of cognitive complexity. For example, evaluating whether a formula from a textbook can be applied to solve a math problem is cognitively a more complex skill compared to recalling that same formula from a textbook. However, a learner needs to be able to recall the formula first, and then evaluate its utility in the context of a genuine problem, i.e., low-order skills are precursors to high-order skills [1, 12]. To define learning objectives at different skill levels, educators often use educational taxonomies (e.g., Bloom's [1, 4], Gagne's [12], and Jensen's [16]). For instance, over decades educators have widely utilized Bloom's taxonomy for Cognitive Objectives [1] to define learning objectives, as this framework can account for a broad range of learning objectives and provide means for evaluating learner achievements relative to those objectives [17]. Bloom's taxonomy consists of six levels of cognitive skills that include 3 low-order (*remember*, *understand*, and *apply*) and 3 high-order cognitive skills (*analyze*,

*Corresponding authors.

Y. Li, M. Rakovic, B. X. Poh, D. Gasevic, and G. Chen. Automatic classification of learning objectives based on Bloom's taxonomy. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 530–537, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852191>

evaluate, and *create*).

Although Bloom's taxonomy has been regarded as a helpful pedagogical framework [19], many educators have difficulties developing learning objectives appropriate to the levels specified in Bloom's taxonomy [21]. This is because they need to consider the progression of learners' skills with learning content and also take into account dependencies between learning objectives, e.g., a learner must be able to define and explain a math formula before applying it [19]. These difficulties may lead to subsequent challenges in measuring learners' progress, i.e., difficulties to determine whether a learner has progressed to upper levels of Bloom's taxonomy [21]. To ensure learning objectives educators create can be mapped to Bloom's taxonomy levels properly, the educators often need support from educational experts [13], which are not easily available in many departments in higher education. Everything considered, the process of developing well-articulated learning objectives to support teaching, learning and assessment activities is usually time- and resource-consuming.

To remedy this challenge and help educators determine the level of each learning objective they create according to Bloom's taxonomy, in the present study, we explored the possibility of using state-of-the-art natural language processing, machine learning and deep learning techniques to automatically classify learning objectives. To date, researchers have developed a few computational models for automatic classification of different types of educational texts based on cognitive levels in Bloom's taxonomy, including exam questions (e.g., [6, 15, 25, 40]), participants' contributions to discussion forums (e.g., [11]), and learning outcomes (e.g., [36]). Researchers have demonstrated a considerable classification accuracy of these models. However, even though the classification of learning objectives based on Bloom's taxonomy has been recognized as an important problem, to our knowledge, there has yet to be developed a classification system that accurately automatizes this work. To address this gap, we obtained and manually annotated 21,380 course learning objectives from 5,558 courses from all the 10 constituent faculties at an Australian university, and applied both machine learning and deep learning models to construct classifiers to automatically identify the cognitive levels of these learning objectives.

2. RELATED WORK

2.1 Bloom's Taxonomy

Bloom's taxonomy [1, 4] was originally introduced to reduce educators' labor when preparing the materials for annual comprehensive examinations. The taxonomy proposes six hierarchically-arranged levels of cognition: remember, understand, apply, analyze, evaluate, and create. These levels reflect the cognitive complexity of a learning objective or an assessment question [15]. In particular, remember, understand, and apply are considered low-order, whereas analyze, evaluate, and create are considered high-order cognitive skills. Mastering a skill at a higher level is dependent upon mastering a prerequisite skill or a group of prerequisite skills at lower levels in Bloom's taxonomy. Due to its well-developed structure, educational researchers and practitioners have widely utilized Bloom's taxonomy for both research and instructional purposes, including the classifica-

tion of learning objectives [36]. We use Bloom's taxonomy as a theoretical framework to guide this study.

2.2 Automated Analysis of Educational Texts Based on Bloom's Taxonomy

Despite its educational promises, the use of Bloom's taxonomy is usually not straightforward. Many educators struggle to manually classify instructional and assessment activities, specify the knowledge associated with each level in Bloom's taxonomy, and measure student progress accordingly ([21]). To overcome these challenges and facilitate instructional activities, researchers have created several computational systems for the automated classification of educational texts based on Bloom's taxonomy. Wen-Chih et al. [5] developed a keyword-based system to automatically classify teachers' questions into different cognitive levels of Bloom's taxonomy. For this purpose, the authors developed a dictionary of keywords mapped to the corresponding cognitive levels of Bloom's taxonomy. The classification system developed in this way achieved a considerable accuracy of 75% in identifying questions at the *remember* level, whereas the system's performance in identifying questions at other levels was noticeably lower (25% – 59%). Amali et al. [28] developed several models to automatically classify exam questions into cognitive levels in Bloom's taxonomy. The models included a rule-based part-of-speech classifier, support vector machine, naive Bayes and K-nearest Neighbor classifiers with word vectors as inputs. Similar to [5], the models performed best in identifying exam questions at the *remember* level (87% – 100%) but achieved a lower overall performance (60% – 72%). The authors further created an ensemble model that achieved 82% overall accuracy by combining the four models. Jayakodi et al. [15] utilized semantic similarity algorithms to develop a rule-based classifier that identifies a cognitive level of an exam question according to Bloom's taxonomy. This system achieved a classification accuracy up to 0.70 in identifying a correct cognitive level for an exam question. Similarly, Echeveria et al. [11] computed TF-IDF features in student discussion posts as input for a rule-based classifier that categorizes a post into one of the levels of Bloom's taxonomy. The authors reported an accuracy of nearly 0.77. Waheed et al. [40] and Mohammed et al. [25] developed a group of supervised machine learning models to classify open-ended questions according to Bloom's taxonomy. The authors computed a variety of linguistic features from question text, e.g., TF-IDF [32] and word2vec [24]. Whereas these supervised machine learning models were trained using relatively small datasets (i.e., less than 1,000 questions), most of them achieved a substantial classification performance with their F1 scores ranging between 0.70 and 0.90.

In line with the increased use of deep learning methods in educational research over the past few years, James et al. [44] utilized BERT [10], a pre-trained language model, to classify educational questions relative to Bloom's taxonomy in a cognitive domain. The models performed well in identifying questions at the levels that were frequent in the dataset (*remember*, *understand*, *analyze* – achieving 82.61% accuracy), whereas the identification of questions at less frequent levels (*apply*, *evaluate*, *create*) remained a challenge (59.2% accuracy with all cognitive levels included). This study demonstrated the potential of deep learning methods to assist educators in determining Bloom's cognitive levels

of educational questions if a sufficiently big pool of questions at all cognitive levels is available to train the deep learning models. Further, Sarang et al. [36] utilized the pre-trained language model Wiki Word Vectors to generate word embeddings for learning objectives and assessment questions. The word embeddings were used as input to the Long Short Term Memory (LSTM) model classifying learning objectives and assessment questions into different levels of Bloom’s taxonomy. This model achieved a weighted average F1 score of 0.73 in correctly classifying learning objectives and a macro-average F1 score of 0.82 in correctly classifying the assessment questions. Moreover, to our knowledge, this study is the first to automatically classify learning objectives (among other forms of educational texts) according to Bloom’s taxonomy and the considerable classification performance reported in the study encourages further research. Overall, the classification models developed to date promise to provide at-scale support to educators who aim at categorizing educational texts, e.g., discussion forum posts, assessment questions, and, more recently, learning objectives, based on cognitive levels of Bloom’s taxonomy.

Although researchers have begun increasingly harnessing the automated text analysis methods to classify different educational texts based on Bloom’s taxonomy, only a small group of researchers has considered exploring the possibility of using these methods to automatically classify learning objectives, despite the challenges documented that many educators report when attempting to manually classify learning objectives according to the cognitive levels of Bloom’s taxonomy [21]. Additionally, all of the relevant studies that we found assumed each piece of text to belong to only one cognitive level while the possibility existed for learning objectives to have more cognitive levels as educators could have combined several learning objectives in one sentence. We also note that practical challenges in obtaining a large, manually labeled dataset, with a sufficient number of learning objectives at each of the six levels of Bloom’s taxonomy to train the classification models, could have been an important obstacle to this line of research [36]. To address these gaps, we gathered a large number of authentic learning objectives across different university courses and manually labeled each learning objective with at least one cognitive level in Bloom’s taxonomy. Next, we developed classification models based on machine learning and deep learning methods to automatically classify learning objectives. More specifically, we attempted to answer the following **research question**: *To what extent can machine learning and deep learning classifiers accurately classify a learning objective into the cognitive levels of Bloom’s taxonomy?*

3. METHODOLOGY

3.1 Data Collection and Labeling

We collected 21,380 learning objectives publicly available from 5,558 courses provided by the 10 faculties at an Australian university in 2021. To collect the data, we developed a web scraper using Python to automatically parse the content of the available course web pages to obtain learning objectives of a course.

One human coder, who had previously received training on Bloom’s taxonomy, manually categorized the learning objectives into their corresponding cognitive levels of Bloom’s

taxonomy. Some learning objectives were categorized into more than one cognitive level, i.e., 2,325 learning objectives were labeled with two cognitive levels, 280 with three and 2 with four levels. We provide a sample of coded learning objectives in Table 1.

To ensure the reliability of data labeling, we included a second human coder trained on Bloom’s Taxonomy in cognitive domain who randomly selected 30% of the learning objectives labeled by the first coder and independently labeled those learning objectives. The two coders achieved a substantial inter-coder agreement (Cohen’s κ 0.63), according to the recommendations provided in [22]. The two coders discussed the labeling disagreement cases between them and found out that the major source of disagreement was because many learning objectives from the low-cognition category *remember* were wrongly categorized as the high-cognition category *apply*. The coders revised the corresponding labels in the entire dataset accordingly which increased inter-coder agreement on the 30% of sample to 0.80, measured with Cohen’s κ . We proceeded with feature engineering and model development using the labeled dataset to answer our research question. The detailed descriptive statistics are provided in Table 2.

3.2 Classification Models

To answer our research question, we developed and examined six classification models. Of these, five models were based on traditional machine learning algorithms, support vector machine (SVM), logistic regression (LR), naive Bayes (NB), random forest (RF) and XGBoost. These algorithms have been widely utilized for text classification tasks in educational research (for an overview see [35]). Moreover, inspired by the increasing use of deep learning approaches in educational research over the past few years (e.g., [7, 9, 14]), we developed deep learning classifiers based on the BERT pre-trained language model. Specifically, we coupled the pre-trained BERT sequence classifiers with a single layer for classification and trained the model using the data we collected.

Recall that, in our collected dataset, each learning objective can be assigned with either only one cognitive label or multiple labels at the same time. Correspondingly, we could tackle the classification task by using two different methods. The first method is to construct a binary classifier for each cognitive level in Bloom’s taxonomy (e.g., those labeled as *remember* vs. those not), and the second method is to build one multi-class multi-label classifier for all the cognitive levels in Bloom’s taxonomy (i.e., identifying all the cognitive labels specific to a learning objective). Given that part of our research goal was to shed light on the best way to tackle this problem, we implemented both methods for comparison purposes. More specifically, we used each of the models described above to construct a binary classifier for each cognitive level in Bloom’s taxonomy, i.e., we constructed a total of 36 binary classifiers. Then, we constructed two multi-class multi-label classifiers (i.e., a Random Forest model and a BERT model), as these two models have been demonstrated effective in tackling multi-class multi-label classification problems in previous research ([27, 37, 39, 42, 43]).

Table 1: Example of learning objectives categorized into the cognitive levels in Bloom’s taxonomy.

Learning Objective Examples	Labels
Recognise the key role that human factors play in the leadership and development of a highly functional perioperative team.	Remember
Describe the general characteristics of the modern X-ray system used in clinical practice, including scientific principles, and production of the digital image.	Understand
Apply research skills to operate effectively as a member of a research project team.	Apply
Identify an issue of relevance to the practice of perioperative medicine capable of further investigation and research within the context of a capstone project.	Analyze
Ability to articulate critical interpretations of dramatic texts and processes in systematic written argument.	Evaluate
A capacity to design, manage, and carry out a research project.	Create
Analyze and apply contemporary management theory and research to current organizational issues.	Apply & Analysis
Assess and synthesise diverse information about up-to-date information and knowledge management systems market and how to use implementation strategies to maximise their strengths and minimise their weaknesses.	Evaluate & Create

Table 2: Descriptive statistics of the Learning Objective (LO) dataset.

	Total	Remember	Understand	Apply	Analyze	Evaluate	Create	Multi-Label
# Total LOs	21,380	886	5,079	5,074	2,311	2,468	2,955	2,607
# Avg. words per LO	17.81	16.14	17.42	18.55	16.68	16.64	16.27	21.52
# Avg. unique words per LO	15.75	14.59	15.33	16.40	15.05	14.91	14.74	18.25

3.3 Study Setup

3.3.1 Data Pre-processing

Prior to conducting any experiments, we randomly split the dataset in an 80:20 ratio, i.e., 80% of data was used as a training set and 20% of data was used as a testing set. We used these same datasets across all classification tasks to ensure fair comparisons between different models.

The textual data was initially pre-processed in the same fashion for both the machine learning models and the BERT-based deep learning model by converting them to lower-case. We extracted multiple features that had been proven to be useful not only for educational forum post classifications [35], but also in other studies sharing a similar context to ours ([25], [32]) to empower the five conventional machine learning models described above. In particular, we computed a group of features in n-gram form, including unigrams (1,000 most frequent excluding stopwords) and bigrams (1,000 most frequent excluding stopwords); TF-IDF features (1,000 most frequent excluding stopwords); automated readability index [33] for each learning objective; and 93 features derived from the LIWC dictionary [31] reflecting a frequency of different psychologically meaningful words, e.g., cognitive processes, function words, words reflecting summary, relativity and time orientation, leading to a total of 3,094 features. For our BERT-based deep learning model, unlike some previous studies ([18, 25, 28]) where the researchers used word2vec to generate word embeddings, we employed BERT-uncased shared by HuggingFace [41] to generate word embeddings, because BERT generated embeddings had been proven to be capable of capturing con-

textual information and properties at the sentence level (in this study, the learning objective level) [23].

3.3.2 Model Implementation

To implement and examine the five conventional machine learning models, we utilized the Python package Scikit-Learn [30] to develop naive Bayes, logistic regression, support vector machine, and random forest classifiers, and the package XGBoost [8] to develop the XGBoost classifier. We performed hyper-parameter tuning with 3-fold cross-validation on these models using grid search in order to find the most suitable parameters for our models. F1 score was used as the evaluation metric when performing hyper-parameter tuning. The details of the parameters were all documented in our source code and would be open-sourced together with the data collected in this study, and thus be made available to other researchers for replications¹

For the BERT-based model, we applied the BERT-uncased shared by HuggingFace [41]. The model included 12 hidden layers, each with 768 neurons. The vocabulary size was 30,522 and the dropout rate was 0.1. For binary sequence classifiers, the number of output neurons is 2, each predicting the probability of the text belonging to different classes (0 and 1 as class labels). Therefore, we applied a softmax function on these probabilities to find the corresponding class labels for the texts. For the multi-class multi-label sequence classifier, the number of output neurons is 6, predicting the probabilities of the text belonging

¹<https://github.com/SteveLEEEEEE/EDM2022CLO.git>

to the six cognitive levels. Thus, we used a sigmoid function on these probabilities and set the probability threshold to 0.50 to find their predicted class labels. The entire BERT models were fine-tuned without freezing the parameters in any layers for all experiments.

3.3.3 Model Training

The training data were used to train all the machine learning models without further splitting. However, for our deep learning models, the 80% training data were further split with 80% being training set and 20% being validation set. The batch size was set to 64 for all the deep learning models and the number of epochs was set to 3. Early-stopping was applied in order to avoid over-fitting. When the F1 scores stopped improving on 10 consecutive validations, the training terminated and the model weights rolled back to the best performing one.

3.3.4 Evaluation Metrics

We evaluated the performance of the classification models by computing the following performance metrics: accuracy, Cohen’s κ , Area Under the ROC Curve (AUC), and F1 score. To find out the categorical performance on the multi-class multi-label classifiers, we separated the classification results for each of the cognitive levels on testing data and made comparisons with the humanly assigned ones to find out their individual accuracy, Cohen’s κ , AUC, and F1 scores.

4. RESULTS

Our results provide evidence that it is possible to develop highly accurate supervised machine learning and deep learning models to classify learning objectives into skill levels based on Bloom’s taxonomy, answering our research question. In particular, the high-performing models included those based on SVM, RF, XGBoost and BERT (Table 3). All of the high-performing models achieved Cohen’s κ score spanning between 0.79 and 0.93, while the prediction accuracy of these models spanned between 0.92 and 0.99, i.e., the models can accurately classify at least 92% of learning objectives into a corresponding skill level of Bloom’s taxonomy. Equally importantly, the F1 scores of the high-performing models were between 0.83 and 0.95, indicating a high precision and recall achieved by SVM, RF, XGBoost, and BERT in identifying each of the six cognitive levels in Bloom’s taxonomy. We note that the binary BERT models outperformed all the binary machine learning models observed. These models achieved an outstanding classification performance, as measured by Cohen’s κ (0.87 to 0.93), accuracy (0.96 to 0.99), and F1 scores (0.88 to 0.95). The classification performance of other models observed in this study (i.e., naive Bayes and logistic regression) was noticeably lower, e.g., with Cohen’s κ typically not exceeding 51% for naive Bayes and 73% for logistic regression.

Furthermore, by comparing the performance between binary and multi-class multi-label random forest classifiers, it is evident that binary random forest classifier outperformed the multi-class multi-label one in most cases except for *understand*. Meanwhile, the multi-class multi-label BERT-based classifier performed better than all the binary and multi-class multi-label machine learning models from the same cognitive level, but rarely outperformed binary BERT-based

classifiers in terms of the prediction performance. The exception was the cognitive level *evaluate* where binary BERT-based classifier achieved a similar but slightly lower performance than the multi-class multi-label BERT-based classifier (i.e., 0.001 difference in Cohen’s κ and F1 score).

5. DISCUSSION

5.1 Interpretation of the Results

Many educators across a range of disciplines develop learning objectives for their courses based on Bloom’s taxonomy for Cognitive Objectives [1]. Even though Bloom’s taxonomy has been widely deemed a useful pedagogical framework [19], educators often find it challenging and tedious to develop learning objectives to describe cognitive skills at different levels of Bloom’s taxonomy [21]. To remedy this issue, in this study, we explored whether machine learning and deep learning methods can be used to develop the classification model that can automatically classify a learning objective into appropriate cognitive level in Bloom’s taxonomy. Overall, our results indicated that three traditional machine learning models, i.e., support vector machine, random forest, and XGBoost, and one deep learning model based on BERT, may be the viable approaches towards solving this problem.

The four high-performing classification models achieved considerable performance not only relative to commonly accepted standards in discourse analysis [3], but these models also outperformed the models from prior research that targeted similar classification tasks. Importantly, all the models performed well in correctly classifying learning objectives at each level of Bloom’s taxonomy. Given that both conventional machine learning (e.g., [5, 28]) and deep learning approaches (e.g., [44]) have been documented to perform poorly in classifying educational texts into higher-order cognitive levels of Bloom’s taxonomy (e.g., *analyze*, *evaluate*, and *create*), the results of our study add to the body of knowledge in educational research showing that advanced conventional machine learning and deep learning models trained on a large corpus of educational textual data can provide useful classifications across all the levels in Bloom’s taxonomy.

Moreover, our findings resonate with prior research showing that deep learning models can provide a more accurate classification results in educational classification tasks, compared to conventional machine learning algorithms [35]. We also note that, given the performance scores the naive Bayes classifier consistently achieved across the six tasks in our study, it appears that this classifier may be the least preferable algorithm for classification tasks based on Bloom’s taxonomy, corroborating evidence provided in [29] where the authors pursued the question classification task based on Bloom’s taxonomy and found that naive Bayes under-performed other classifiers in this task.

Last, we observed that, though multi-class multi-label classifiers achieved satisfactory performance, binary classifiers using the same model (i.e., BERT) still attained better performance. This might be mainly because that, while multi-class multi-label classifiers tried to minimize the overall errors across different cognitive levels during the model training process, binary classifiers tended to focus comprehensively on minimizing the errors on a single category. There-

Table 3: Classification Performance of the binary and multi-class multi-label (MCML) classifiers, i.e., Support Vector Machine (SVM), Naive Bayes (NB), Logistic Regression (LR), Random Forest (RF), XGBoost, and the BERT-based classifier. The best results are in bold for each evaluation metric in each level of Bloom’s taxonomy.

Methods		Remember				Understand				Apply			
		Acc.	Cohen’s κ	AUC	F1	Acc.	Cohen’s κ	AUC	F1	Acc.	Cohen’s κ	AUC	F1
Binary Classifiers	NB	0.640	0.111	0.716	0.198	0.642	0.327	0.716	0.581	0.778	0.507	0.781	0.668
	SVM	0.982	0.827	0.923	0.837	0.922	0.801	0.891	0.855	0.923	0.805	0.890	0.858
	LR	0.960	0.485	0.681	0.503	0.891	0.714	0.839	0.787	0.896	0.726	0.837	0.793
	RF	0.983	0.830	0.892	0.839	0.920	0.793	0.880	0.847	0.936	0.837	0.904	0.881
	XGBoost	0.981	0.820	0.916	0.830	0.928	0.818	0.900	0.867	0.938	0.844	0.914	0.887
	BERT	0.987	0.871	0.916	0.878	0.971	0.926	0.959	0.947	0.961	0.904	0.951	0.931
MCML Classifiers	RF	0.982	0.809	0.989	0.818	0.927	0.811	0.970	0.860	0.921	0.794	0.970	0.847
	BERT	0.984	0.848	0.988	0.856	0.955	0.889	0.982	0.920	0.951	0.877	0.976	0.912
Methods		Analyze				Evaluate				Create			
		Acc.	Cohen’s κ	AUC	F1	Acc.	Cohen’s κ	AUC	F1	Acc.	Cohen’s κ	AUC	F1
Binary Classifiers	NB	0.549	0.183	0.684	0.392	0.596	0.234	0.703	0.447	0.676	0.300	0.743	0.474
	SVM	0.956	0.832	0.897	0.858	0.959	0.861	0.922	0.886	0.942	0.791	0.880	0.825
	LR	0.936	0.732	0.818	0.767	0.920	0.694	0.799	0.739	0.902	0.604	0.762	0.659
	RF	0.961	0.851	0.902	0.874	0.967	0.887	0.932	0.907	0.943	0.792	0.877	0.826
	XGBoost	0.959	0.844	0.903	0.868	0.964	0.878	0.924	0.900	0.944	0.796	0.882	0.829
	BERT	0.975	0.906	0.950	0.922	0.974	0.913	0.954	0.929	0.962	0.866	0.924	0.888
MCML Classifiers	RF	0.951	0.803	0.972	0.831	0.950	0.822	0.981	0.852	0.928	0.715	0.964	0.755
	BERT	0.971	0.890	0.984	0.907	0.974	0.914	0.989	0.930	0.958	0.846	0.971	0.872

fore, with adequate data collected, tackling the problem as multiple binary classification tasks may be a better solution.

5.2 Practical Implications

In this study, we made a first step towards developing future computational tool that can provide at-scale support to instructors, instructional designers, and other educational stakeholders who aim at developing learning objectives well aligned to Bloom’s taxonomy. The system will automatically analyze learning objectives using the classification routines developed in this study. For instance, an instructional designer may submit the list of manually created course learning objectives to this future system and obtain a highly accurate classification of the learning objectives into cognitive levels of Bloom’s taxonomy. Using this information, the instructional designer may determine whether all the learning objectives are provided, relative to course requirements, e.g., “*It looks like I yet to develop a learning objective at the create level. Since this is an advanced writing course, the create learning objectives should be included*” or “*Even though I have created a few learning objectives for the skills at the apply level, my list is missing lower-level learning objectives that represent the corresponding pre-requisite skills*”. Overall, the classifiers developed in this study can be used to automatically diagnose the cognitive levels of learning objectives for courses and educational programs across different higher education institutions.

In addition, coupled with the systems for natural language generation, the classifiers of learning objectives might be further enhanced to automatically generate learning objectives from course content. This, in turn, may reduce time educators dedicate to this task and may mitigate inconsistencies educators introduce among each other when defining learning objectives, e.g., two instructors defining different learn-

ing objectives for the same subject. We also anticipate our work will benefit students by providing means for automatic development of questions of different cognitive levels for self-assessment. For example, automatically generated learning objectives can be further coupled with the systems for automatic question generation to obtain interrogative form for objectives. Questions developed in this way may provide at-scale support to students studying for assessment.

6. LIMITATIONS AND FUTURE WORK

We identified several limitations in this study that may be considered in future research. Firstly, even though all the learning objectives we collected were classified into at least one of the cognitive levels in Bloom’s taxonomy, it is, however, possible that some learning objectives cannot be categorized relative to a cognitive domain but relative to other domains instead, e.g., affective domain [26]. In future research, the learning objectives dataset should be further labeled from other domains, and relevant classifiers should be trained to recognize these types of learning objectives. Secondly, the supervised machine learning and deep learning methods utilized in this study require extensive amounts of labeled data to achieve a highly accurate prediction performance. As preparing such a large-scale dataset can be costly and time-consuming, researchers may consider using semi-supervised machine learning approaches (e.g., semi-supervised Random Forest [20]) or training strategies like active learning [34] to enable more effective and efficient model construction process in the future.

7. REFERENCES

- [1] L. W. Anderson and D. R. Krathwohl. *A taxonomy for learning, teaching, and assessing: A revision of Bloom’s taxonomy of educational objectives*. Longman,, 2001.

- [2] H. L. Andrade. A critical review of research on student self-assessment. In *Frontiers in Education*, volume 4, page 87. Frontiers, 2019.
- [3] R. Artstein and M. Poesio. Inter-coder agreement for computational linguistics. *Computational Linguistics*, 34:555–596, 2008.
- [4] B. S. Bloom et al. Taxonomy of educational objectives. vol. 1: Cognitive domain. *New York: McKay*, 20(24):1, 1956.
- [5] W.-C. Chang and M.-S. Chung. Automatic applying bloom’s taxonomy to classify and analysis the cognition level of english question items. In *2009 Joint Conferences on Pervasive Computing (JPCP)*, pages 727–734, 2009.
- [6] G. Chen, J. Yang, C. Hauff, and G.-J. Houben. Learningq: a large-scale dataset for educational question generation. In *Twelfth International AAAI Conference on Web and Social Media*, 2018.
- [7] J. Chen, J. Feng, X. Sun, and Y. Liu. Co-training semi-supervised deep learning for sentiment classification of mooc forum posts. *Symmetry*, 12(1):8, 2020.
- [8] T. Chen and C. Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD ’16, pages 785–794, New York, NY, USA, 2016. ACM.
- [9] B. Clavié and K. Gal. Edubert: Pretrained deep language models for learning analytics. *arXiv preprint arXiv:1912.00690*, 2019.
- [10] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [11] V. Echeverría, J. C. Gomez, and M.-F. Moens. Automatic labeling of forums using bloom’s taxonomy. In *International Conference on Advanced Data Mining and Applications*, pages 517–528. Springer, 2013.
- [12] R. M. Gagné, R. M. Gagné, et al. *Conditions of learning and theory of instruction*. Holt, Rinehart and Winston, 1985.
- [13] R. Gluga, J. Kay, R. Lister, Simon, and S. Kleitman. Mastering cognitive development theory in computer science education. *Computer Science Education*, 23(1):24–57, 2013.
- [14] S. X. Guo, X. Sun, S. X. Wang, Y. Gao, and J. Feng. Attention-based character-word hybrid neural networks with semantic and structural information for identifying of urgent posts in mooc discussion forums. *IEEE Access*, 7:120522–120532, 2019.
- [15] K. Jayakodi, M. Bandara, I. Perera, and D. Meedeniya. Wordnet and cosine similarity based classifier of exam questions using bloom’s taxonomy. *International Journal of Emerging Technologies in Learning*, 11(4), 2016.
- [16] A. Jensen. Varieties of individual differences in learning. in, rm gagne. *Learnin~ and Individual Differences*. Columbus, Ohio: Merrill Books, 1967.
- [17] P. C. Kyllonen and V. J. Shute. Taxonomy of learning skills. Technical report, UNIVERSAL ENERGY SYSTEMS INC DAYTON OH, 1988.
- [18] M. Laddha, V. Lokare, A. Kiwelekar, and L. Netak. Classifications of the summative assessment for revised bloom’s taxonomy by using deep learning. *International Journal of Engineering Trends and Technology*, 69:211–218, 03 2021.
- [19] M. B. Larson and B. B. Lockee. *Streamlined ID: A practical guide to instructional design*. Routledge, 2019.
- [20] C. Leistner, A. Saffari, J. Santner, and H. Bischof. Semi-supervised random forests. In *2009 IEEE 12th international conference on computer vision*, pages 506–513. IEEE, 2009.
- [21] S. Masapanta-Carrión and J. Á. Velázquez-Iturbide. A systematic review of the use of bloom’s taxonomy in computer science education. In *Proceedings of the 49th acm technical symposium on computer science education*, pages 441–446, 2018.
- [22] M. L. McHugh. Interrater reliability: the kappa statistic. *Biochemia medica*, 22(3):276–282, 2012.
- [23] A. Miaschi and F. Dell’Orletta. Contextual and non-contextual word embeddings: an in-depth linguistic investigation. In *Proceedings of the 5th Workshop on Representation Learning for NLP*, pages 110–119, Online, July 2020. Association for Computational Linguistics.
- [24] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.
- [25] M. Mohammed and N. Omar. Question classification based on bloom’s taxonomy cognitive domain using modified tf-idf and word2vec. *PLoS one*, 15(3):e0230442, 2020.
- [26] R. W. Morshead. Taxonomy of educational objectives handbook ii: Affective domain. *Studies in Philosophy and Education*, 4(1):164–170, 1965.
- [27] V. Nikolovski, D. Kitanovski, D. Trajanov, and I. Chorbev. Case study: Predicting students objectivity in self-evaluation responses using bert single-label and multi-label fine-tuned deep-learning models. In V. Dimitrova and I. Dimitrovski, editors, *ICT Innovations 2020. Machine Learning and Applications*, pages 98–110, Cham, 2020. Springer International Publishing.
- [28] A. Osadi, N. Fernando, and V. Welgama. Ensemble classifier based approach for classification of examination questions into bloom’s taxonomy cognitive levels. *International Journal of Computer Applications*, 162:975–8887, 04 2017.
- [29] A. Osman and A. A. Yahya. Classifications of exam questions using natural language syntatic features: A case study based on bloom’s taxonomy. In *Proc. 6th Int. Arab Conf. Quality Assurance Higher Edu.*, 2016.
- [30] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [31] J. W. Pennebaker, R. L. Boyd, K. Jordan, and

- K. Blackburn. The development and psychometric properties of liwc2015. Technical report, 2015.
- [32] C. Sammut and G. I. Webb. *Encyclopedia of machine learning*. Springer Science & Business Media, 2011.
- [33] R. Senter and E. A. Smith. Automated readability index. Technical report, Cincinnati Univ OH, 1967.
- [34] B. Settles. Active learning literature survey. 2009.
- [35] L. Sha, M. Rakovic, Y. Li, A. Whitelock-Wainwright, D. Carroll, D. Gašević, and G. Chen. Which hammer should i use? a systematic evaluation of approaches for classifying educational forum posts. *International Educational Data Mining Society*, 2021.
- [36] S. Shaikh, S. M. Daudpotta, and A. S. Imran. Bloom’s learning outcomes’ automatic classification using lstm and pretrained word embeddings. *IEEE Access*, 9:117887–117909, 2021.
- [37] S. Sharma and D. Mehrotra. Comparative analysis of multi-label classification algorithms. In *2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC)*, pages 35–38, 2018.
- [38] H. Sullivan and N. Higgins. *Teaching for competence*. ERIC, 1983.
- [39] T. Tang, X. Tang, and T. Yuan. Fine-tuning bert for multi-label sentiment analysis in unbalanced code-switching text. *IEEE Access*, 8:193248–193256, 2020.
- [40] A. Waheed, M. Goyal, N. Mittal, D. Gupta, A. Khanna, and M. Sharma. Bloomnet: A robust transformer based model for bloom’s learning outcome classification. *arXiv preprint arXiv:2108.07249*, 2021.
- [41] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, and J. Brew. Huggingface’s transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771, 2019.
- [42] R. Yarullin and P. Serdyukov. Bert for sequence-to-sequence multi-label text classification. In W. M. P. van der Aalst, V. Batagelj, D. I. Ignatov, M. Khachay, O. Koltsova, A. Kutuzov, S. O. Kuznetsov, I. A. Lomazova, N. Loukachevitch, A. Napoli, A. Panchenko, P. M. Pardalos, M. Pelillo, A. V. Savchenko, and E. Tutubalina, editors, *Analysis of Images, Social Networks and Texts*, pages 187–198, Cham, 2021. Springer International Publishing.
- [43] H. M. Zahera. Fine-tuned bert model for multi-label tweets classification. In *TREC*, 2019.
- [44] J. Zhang, C. Wong, N. Giacaman, and A. Luxton-Reilly. Automated classification of computing education questions using bloom’s taxonomy. In *Australasian Computing Education Conference, ACE ’21*, page 58–65, New York, NY, USA, 2021. Association for Computing Machinery.

Skills Taught vs Skills Sought: Using Skills Analytics to Identify the Gaps between Curriculum and Job Markets

Alireza Ahadi
University of Technology
Sydney
PO Box 123 Broadway, Ultimo,
2007, Australia
alireza.ahadi@uts.edu.au

Marian-Andrei Rizoiu
University of Technology
Sydney
PO Box 123 Broadway, Ultimo,
2007, Australia
marian-
andrei.rizoiu@uts.edu.au

Kirsty Kitto
University of Technology
Sydney
PO Box 123 Broadway, Ultimo,
2007, Australia
kirsty.kitto@uts.edu.au

Katarzyna Musial
University of Technology
Sydney
PO Box 123 Broadway, Ultimo,
2007, Australia
katarzyna.musial-
gabrys@uts.edu.au

ABSTRACT

Higher education often aims to create job-ready graduates. Thus, the skills and knowledge taught in professional degrees are expected to align with the needs of the labor market. However, the dynamic nature of the job market makes it challenging to ensure that this alignment occurs. In this study, we show how Skills Analytics can be used to identify critical skills in the workforce, mapping these to the curriculum offerings of a university. This enables us to identify *skill gaps* between what is taught and what is needed in the job market. Methods are presented that allow universities to test the alignment of their curriculum offerings with the job market. Where gaps are identified, this would enable universities to update their curriculum more rapidly to produce graduates equipped with up-to-date skills required by the local job market. Our contributions include: a new method for ranking skills in curricula based on their relative importance in the job market; and proof of concept methods to find skills gaps between curriculum offerings and an identified job market that can lead to curriculum redesign and enhancements.

Keywords

Curriculum analytics, Skills Analytics, Skills gap

1 Introduction and Prior Research

As modern society increases in complexity, higher education is often seen as a necessity for achieving success in the workplace [7]. While this claim is sometimes challenged [5], tertiary training is still often regarded as the entry-level qual-

ification for a professional job [21]. While many claim that higher education should be aiming not just to develop work readiness but also to support people in becoming critical thinkers able to contribute to the betterment of our society [11], we increasingly see both government and employers pushing universities to prepare graduates who can make an immediate contribution in the workplace [16, 19].

How do we know that a university degree adequately prepares a student for the labor market? Over the years, a wide range of computational models have been developed to identify and/or catalog the skills and knowledge associated with a course or degree program. One popular method arises from the curation of ontologies describing a set of skills identified as critical by professional associations, employers, or some form of project. For example, the Association for Computing Machinery (ACM) has been recommending curriculum for various computer science disciplines for decades. The most recent Computing Curriculum (CC2020), includes recommendations about curriculum covering: Computer Engineering; Computer Science; Cybersecurity; Information Systems; Information Technology; Software Engineering; and an emerging Data Science category. These standards traditionally represent the curriculum for a job role as a Body of Knowledge (BoK) consisting of a set of knowledge areas (KAs), both of which contain about ten more specialized knowledge units (KUs) described by a short document. Such formal ontologies and mappings of KAs are often applied in EDM to support various Intelligent Tutoring Systems (ITS) [10].

Ontologies are often leveraged by semantic web technologies to represent curriculum [22, 12, 1], an approach that can help to ensure interoperability of curriculum data between institutions [9]. Gasmı and Bouras [13] demonstrate that it is possible to compare the competencies required by industry with those taught by education using a semantic web ontology and then propose an inference engine that can be used to match students or curriculum to jobs. However, to the best of our knowledge their proposed system was never deployed at scale.

An alternative, less manual way in which to link curriculum

A. Ahadi, K. Kitto, M.-A. Rizoiu, and K. Musial. Skills taught vs skills sought: Using skills analytics to identify the gaps between curriculum and job markets. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 538–542, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853121>

to skills involves the use of Natural Language Processing (NLP). A number of different papers have extracted skills from curriculum documents using simple term frequency-inverse document frequency (TF-IDF) approaches [23, 14, 17]. More advanced methods have used simplified-supervised Latent Dirichlet Allocation (ssLDA) to compare computer science (CS) related curriculum offerings across 10 different universities in the United States [20]. This method was used to model a complete curriculum pathway by finding the center of all points corresponding to a syllabus in a degree program. However, while interesting computational approaches, these methods only work because the detailed ACM curriculum documentation has been carefully curated. This is a significant impost, and is not one that has been repeated by many other sectors. As such, it is not a method that scales to an entire university curriculum.

Various nations and even the private sector, have attempted to address the problem of field specificity by developing national standards for the skills, competencies, knowledge and abilities required for the entire workforce. Examples include: the USA based Occupational Information Network Program (O*NET); Singapore’s Skills Framework; and the classification of European Skills, Competences, Qualifications, and Occupations (ESCO). In the same vein, numerous companies are working to provide technical solutions that support people in becoming more employable, from large companies like Microsoft who are using the knowledge graph acquired from their acquisition of LinkedIn, to start ups and Small to Medium Enterprises (SMEs) such as Faethm, who have recently been acquired by Pearson. Similarly, EMSI-Burning Glass technologies (EBG) collect data by web scraping over 40,000 distinct job boards and company websites, and provides detailed daily information about labor and skill demand posted online.

Multiple studies have taken advantage of these skill taxonomies to explore labor market dynamics and the skills demanded by employers. For example, Clemens et al. [6] investigated whether minimum wage increases result in substitution from lower-skilled to slightly higher-skilled labor. Brüning and Mangeol [4] investigated the geographical variation of employer demand for graduate skills within and among occupations. Interestingly, these skills taxonomies are sometimes provided with services to parse documents and return a list of skills. This type of functionality was leveraged in a recent LAK paper by Kitto et al. [18] that defined a notion of *Skills Analytics* to support the Recognition of Prior Learning (RPL) between two institutions, but this same type of mapping is in principle possible between a student and a job, and in an EDM poster by the same group to perform some preliminary profiling of the entire curriculum offered across one university [15]. However, a problem remains: not all skills contribute equally to the employability of a graduate. Which skills are the most important?

Intriguingly, Dawson et al. [8] used Burning Glass data, coupled with a measure from labor economics, Revealed Comparative Advantage (RCA) [3, 2] to calculate the relative importance of skills that are associated with job ads of different occupations. This index is based on Ricardian trade theory and is widely used in international economics for calculating the relative advantage or disadvantage of a certain country in a certain class of commodity as evidenced by trade flows. RCA works for skills in the same way, enabling the calculation of the *advantage* conferred by a skill to a per-

son in a particular context. As such, the RCA provides us with a promising method for analyzing the relative importance of a specific skill to a job, or to a curriculum offering, or indeed, to a graduate when they attempt to find a job. However, to the best of our knowledge this metric has yet to be applied to the problem of finding gaps in an institution’s curriculum that would reduce a graduate’s potential future employability. This is the gap in the literature addressed by our paper, which proceeds by asking the following Research Question:

RQ: *How can we identify skills gaps between content taught at an institution (i.e. its curriculum offerings) and the requirements of a local job market?*

We adopt an approach based upon *Skills Based Curriculum Analytics* [18] to link university curricula with job market data. Extending the methodology proposed by Dawson et al. [8], we present an approach that enables the comparison of the skills taught within a specific degree program offered by a university with the skills sought by employers for a set of occupations targeted by that degree program. A proof of concept is presented, which contributes: (i) The use of a measure from labor market economics, Revealed Comparative Advantage (RCA), to weight skills according to the advantage they confer to a specific context; (ii) A method for evaluating the skills gaps that occur between a curriculum offering (i.e. identified degree program) and specific occupations targeted by that degree program; (iii) A method for measuring the criticality of any skills gaps, which can be used by university decision makers to prioritize curriculum redevelopment.

2 Methods

2.1 Definitions

For the sake of clarity, we will define here the precise meaning of different terms used across this paper. We will denote a *course* as a “course of study” or a degree or credential program offered by a University (e.g. a Bachelor of Science or a Master of Information Technology). A *subject* is taken to be a specific unit of study that a student undertakes during a course (e.g. Introduction to Programming, Introductory Calculus). A *job* reflects a one-on-one relationship between an employee and an employer hence a job ad represents a single vacancy. An *occupation* is a group of jobs that share very similar characteristics. For example, “teacher” is an occupation, but there are many different types of teachers, such as special education teachers and biology teachers. We use the term *skill* to denote skills, capabilities and knowledge components that are associated with a subject, course, job, or occupation interchangeably. In this study, we will represent subjects, courses, jobs and occupations as bags of skills where each bag of skills represents the skill set taught in a subject, attained by completing a course, or required for a job or an occupation, respectively.

2.2 Data Preparation

To prepare the skill-based presentation of the entire curricula of our university, we took every subject offered at the University of Technology Sydney (UTS) and extracted the following information from the Curriculum Information System: a general description of each subject, any information available about content taught, learning objectives, gradu-

ate attributes, and a high-level overview of every assessment. Next, this extracted textual data was sent to a parsing service offered by EMSI-Burning Glass Technology (EBG) (i.e. the curriculum parser tool) through API calls for skill tagging. EBG provides a taxonomy of more than 33,000 skills that are curated from job advertisement data. For each API call, the curriculum parser uses NLP to extract a list of skills which are probabilistically correlated to the subject information. This method returned skill lists for 2,747 subjects, resulting in a grand total of 138,455 skill tags. The occupational data used in this pilot study includes a relatively small set of job advertisements posted during 2020 (N = 144k) that represents a total number of 612 occupations covering 78k unique job titles. The mapping of the job ads to the occupation titles and their underlying skills (1.4M skill tags) was carried out by EBG.

2.3 Finding skill gaps between courses and the occupations they target

The procedure used to identify skills gap between curricula and job market is as follows:

- Step 1: Calculate the importance of skills for each subject
- Step 2: Calculate the importance of skills for each job ad
- Step 3: Calculate the pair-wise co-occurrence of skills for each subject
- Step 4: Calculate the pair-wise co-occurrence of skills for each job ad
- Step 5: Calculate the averaged importance of the skills associated with the job ads of each occupation
- Step 6: Calculate the averaged importance of the skills associated with the subjects of each course
- Step 7: Compute the pairwise similarity matrix of all courses and all occupations

For details about the calculations themselves, and how the RCA was implemented, the paper by Dawson et al. [8] should be consulted.

3 Results

Having used the RCA to weight the relative importance of skills, and constructed a similarity matrix, we can now work to identify skills gaps between the occupations targeted by various courses, and the skills that those courses teach. In this section we will demonstrate how these methods might be used in an institutional setting to extract actionable insights capable of informing curriculum development.

3.1 The RCA downgrades common skills

Firstly, we note that the RCA can help us to reduce the relative importance of very general skills. For example, at UTS ‘Creativity’ is returned by the EBG tagger 1,496 times across all courses that return at least one skill (see Table 1 for more examples). While possessing these general skills is *expected*, having them on a resume is unlikely to *distinguish* graduates in the job market. The RCA scoring method downgrades common skills with lower scores. Cross-checking these skills with the O*NET categorization of skills confirms that they tend to belong to the basic skill category. On the other side, skills with a high RCA score were not present in the basic skills category and so represent more *specialized* skills that distinguish individuals with respect to specific jobs. This in-

Table 1: Highly common skills in UTS curricula, with their counts across the 2,747 subjects, and the % column giving the relative frequency.

Skill	Count	%
Research	2368	86%
Teamwork / Collaboration	1788	65%
Writing	1647	59%
Creativity	1496	54%
Planning	1419	51%
Communication Skills	1401	51%
Problem Solving	1356	49%
Project Management	1337	48%
Customer Service	1240	45%
Organizational Skills	1218	44%
Building Effective Relationships	1181	42%
Budgeting	1169	42%
Presentation Skills	1160	42%
Detail-Oriented	1098	39%
Written Communication	1077	39%
Scheduling	1053	38%
Microsoft Excel	1050	38%
Biologics Development	1045	38%
Microsoft PowerPoint	1026	37%
Multi-Tasking	994	36%

dicates that the RCA is largely behaving as expected in its ranking of skills. It is important to note that some analyses might not want to downgrade common skills in this manner, in which case a different weighting method could be used.

3.2 Skill gaps across a collection of courses

With the methods described in Section 2 it is now possible to explore how various courses offered by UTS align with the occupations that they are attempting to prepare their students for. We have selected 10 courses which aim to prepare their students for professional careers (and so should align well to the needs of the job market), and examined their curriculum descriptions to extract target occupations (e.g. “The course prepares students to participate in a variety of emerging careers with the growth of data science — *Data Scientist, Data Engineer...*”). We also selected a small number of occupations not being targeted by any course in our sample. We then performed the analysis represented by Steps 1 to 7 to calculate the average similarity between the bag of skills returned for each course and each of the selected occupations. The results of this analysis are presented in Figure 1. According to this heatmap, the group of Information Technology courses show a relatively small skills gap for the IT related occupations, and a correspondingly high skills gap for the occupations that they are not claiming to prepare the students for. For example, these courses perform fairly poor in preparing graduates for pharmacy related occupations. On the other side, it is evident that the pharmaceutical occupations in fact show a small skills gap with the Master of Pharmacy. One interesting finding of this presentation is the fact that some courses in fact represent a low skills gap with some occupations that they are not targeting. For instance, the Master of Forensic Sciences shows a relatively small skills gap with *Data Scientist* occupation. This is likely due to a strong emphasis upon analytical thinking and some data analysis skills which are used to provide insights into the elements of a crime. This brings another application of the proposed method; to identify potential alternative career pathways for students

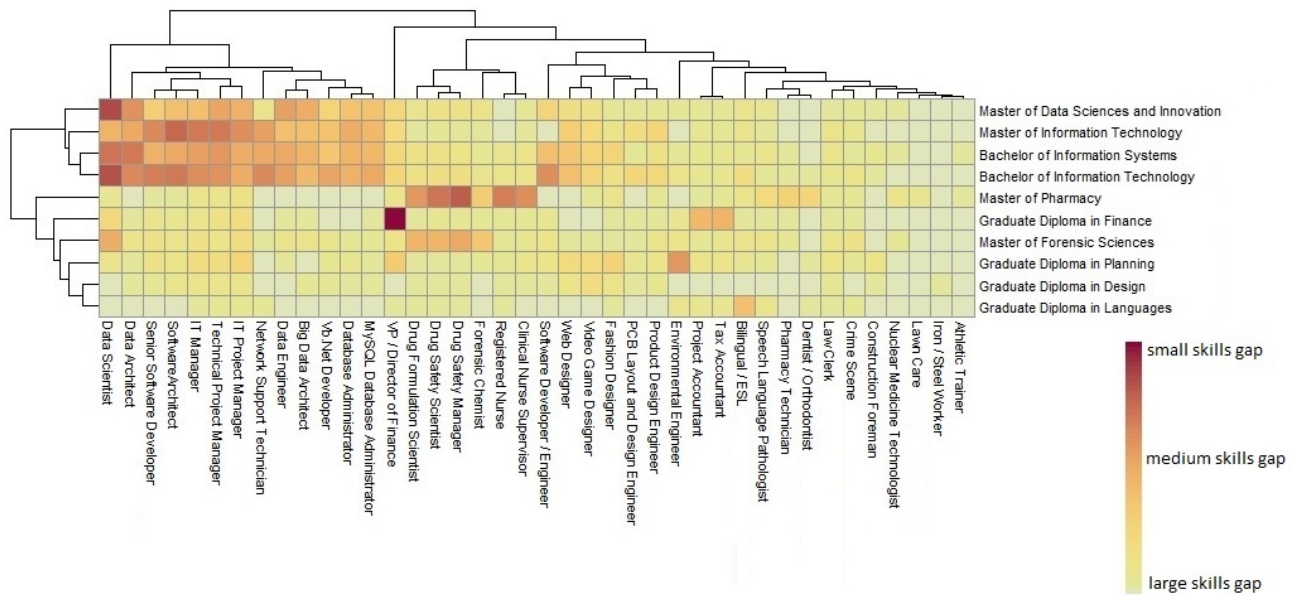


Figure 1: A heat map, representing the skills gap between 50 selected occupations and 10 courses offered at UTS.

beyond their chosen enrollment if they are dissatisfied with their existing pathway. We reserve the examination of such applications of our method for future work.

The hierarchical clustering of the occupations has successfully identified two major clusters, one including IT related occupations, and a second cluster that covers a number sub-clusters representing the other occupation categories chosen (i.e. pharmaceutical, finance, design, language, and forensic). Note in particular, that the occupations to the far right of Figure 1 show a high skill gap. This is a good sign, as they were the occupations chosen which are not targeted by any of the courses we chose to investigate. Given that the similarity calculation ranges from 0 to 1, these results are surprisingly good for two reasons. First, there are 6,684 skills in the occupation space (extremely unique skills are removed) and 3,467 skills in the curriculum space. The fact that there is such overlap between the courses and their intended occupations is a sign that the courses are indeed doing a good job in preparing their students with the skills required in the labor market. Second, as mentioned in Section 2, the occupational data used for this study includes the job advertisements of only one year (2020). This negatively impacts the results of our analysis because it reduces the accuracy of the RCA scores (as a job ad with an unusual skill profile can have a disproportionate influence over the form an occupation takes). Furthermore, the small sample means that some of the skills taught in the course that are in fact required for a given occupation are not identified in the job ads that we had access to. We expect our results to improve with access to a larger job ad dataset.

4 Discussion and Conclusions

Returning to our original research question (in Section 1), this paper has presented a new method that shows clear promise for identifying skill gaps between the content taught by an institution and the job market that it is targeting. The methods proposed in this paper are based on the assumption that relevant textual data can be represented as

a bag of skills. Keeping that notion in mind, our methods can be used in other scenarios as well. For example, this same method could potentially be used when developing a new course or subject to identify significant overlaps in the skills covered by two existing subjects, which may indicate that resources could be more effectively allocated towards the teaching of new content instead. In the same vein, this technique could be adopted for the automated identification of existing competencies, which could be used to identify students who are enrolling in a subject that is unlikely to add much value to their current skill set. While the potential applications of the method presented here are broad, it currently has some limitations. By far the most concerning centers upon the notion of *skillifying*¹ curriculum, occupations, or even a person’s resume or portfolio. We admit that this approach necessarily entails a highly simplified representation of each of these complex entities, and may be critiqued for this reason. However, we feel that the potential utility of the approach in supporting institutions to improve the employability of their graduates justifies this simplification. We believe that RCA can be used for recommending subjects/courses that minimize the skill gaps of each individual with respect to occupation goals that they specify in an interface. Such an interface could also be used to support students in identifying *alternative* occupation goals that they have not yet considered. We reserve these intriguing possibilities for future work.

In conclusion, this paper has provided a proof of concept method for mapping between the skills sought in a local job market, and the skills taught by an institution, an advance that will help to improve student employability in a rapidly changing workforce.

Acknowledgments: We acknowledge the support of Burning Glass in provisioning the API tools and data that were used in this study.

¹See <https://kb.emsidata.com/glossary/skillify/>

References

- [1] M. Al-Yahya, A. Al-Faries, and R. George. Curonto: An ontological model for curriculum representation. In *Proceedings of the 18th ACM conference on Innovation and technology in computer science education*, pages 358–358, 2013.
- [2] A. Alabdulkareem, M. R. Frank, L. Sun, B. AlShebli, C. Hidalgo, and I. Rahwan. Unpacking the polarization of workplace skills. *Science Advances*, 4(7):eaao6030, 2018. doi: 10.1126/sciadv.aao6030.
- [3] B. Balassa. Trade liberalisation and “revealed” comparative advantage 1. *The manchester school*, 33(2): 99–123, 1965.
- [4] N. Brüning and P. Mangeol. What skills do employers seek in graduates?: Using online job posting data to support policy and practice in higher education. 2020.
- [5] T. Chamorro-Premuzic and B. Frankiewicz. Does higher education still prepare people for jobs. *Harvard Business Review*, 7, 2019.
- [6] J. Clemens, L. B. Kahn, and J. Meer. Dropouts need not apply? the minimum wage and skill upgrading. *Journal of Labor Economics*, 39(S1):S107–S149, 2021.
- [7] N. R. Council et al. *Education for life and work: Developing transferable knowledge and skills in the 21st century*. National Academies Press, 2012.
- [8] N. Dawson, M.-A. Rizoiu, and M.-A. Williams. Job transitions in a time of automation and labor market crises. *arXiv e-prints*, pages arXiv–2011, 2020.
- [9] Y. Demchenko, L. Comminiello, and G. Reali. Designing customisable data science curriculum using ontology for data science competences and body of knowledge. In *Proceedings of the 2019 International Conference on Big Data and Education*, pages 124–128, 2019.
- [10] M. Desmarais and R. Baker. A review of recent advances in learner and skill modeling in intelligent learning environments. *User Modeling and User-Adapted Interaction*, 22(1-2):9–38, 2012.
- [11] O. L. U. Enciso, D. S. U. Enciso, and M. d. P. V. Daza. Critical thinking and its importance in education: Some reflections. *Rastros Rostros*, 19(34):78–88, 2017.
- [12] D. Gašević, J. Jovanović, and V. Devedžić. Ontology-based annotation of learning object content. *Interactive Learning Environments*, 15(1):1–26, 2007.
- [13] H. Gasmı and A. Bouras. Ontology-based education/industry collaboration system. *IEEE Access*, 6: 1362–1371, 2017.
- [14] A. Gibson, K. Kitto, and J. Willis. A cognitive processing framework for learning analytics. In *Proceedings of the fourth international conference on learning analytics and knowledge*, pages 212–216, 2014.
- [15] A. Gromov, A. Maslennikov, N. Dawson, K. Musial, and K. Kitto. Curriculum profile: modelling the gaps between curriculum and the job market. In *In Proceedings of The 13th International Conference on Educational Data Mining (EDM 2020)*, pages 610–614, 2020.
- [16] D. Jackson and R. Bridgstock. Evidencing student success in the contemporary world-of-work: Renewing our thinking. *Higher Education Research & Development*, 37(5):984–998, 2018.
- [17] K. Kawintiranon, P. Vateekul, A. Suchato, and P. Punyabukkana. Understanding knowledge areas in curriculum through text mining from course materials. In *2016 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE)*, pages 161–168. IEEE, 2016.
- [18] K. Kitto, N. Sarathy, A. Gromov, M. Liu, K. Musial, and S. B. Shum. Towards skills-based curriculum analytics: can we automate the recognition of prior learning? In *Proceedings of the Tenth International Conference on Learning Analytics & Knowledge*, pages 171–180, 2020.
- [19] T. Pham and D. Jackson. The need to develop graduate employability for a globalized world. In *Developing and Utilizing Employability Capitals*, pages 21–40. Routledge, 2020.
- [20] T. Sekiya, Y. Matsuda, and K. Yamaguchi. Curriculum analysis of cs departments based on cs2013 by simplified, supervised lda. In *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge*, pages 330–339, 2015.
- [21] C. Sin, O. Tavares, and A. Amaral. Accepting employability as a purpose of higher education? academics’ perceptions and practices. *Studies in Higher Education*, 44(6):920–931, 2019.
- [22] K. Verbert, J. Klerkx, M. Meire, J. Najjar, and E. Duval. Towards a global component architecture for learning objects: An ontology based approach. In *OTM Confederated International Conferences” On the Move to Meaningful Internet Systems”*, pages 713–722. Springer, 2004.
- [23] L. S. Xun, S. Gottipati, and V. Shankaraman. Text-mining approach for verifying alignment of information systems curriculum with industry skills. In *2015 International Conference on Information Technology Based Higher Education and Training (ITHET)*, pages 1–6. IEEE, 2015.

DeepIRT with a Hypernetwork to Optimize the Degree of Forgetting of Past Data

Emiko Tsutsumi
University of
Electro-Communications
tsutsumi@ai.lab.uec.ac.jp

Yiming Guo
University of
Electro-Communications
guo_ymzng@
ai.lab.uec.ac.jp

Maomi Ueno
University of
Electro-Communications
ueno@ai.lab.uec.ac.jp

ABSTRACT

Knowledge Tracing (KT), the task of tracing students' knowledge state, has attracted attention in the field of artificial intelligence. Recently, many researchers have proposed KT methods using deep learning to predict student performance on unknown tasks based on learning history data. Especially, the latest DeepIRT reportedly has high predictive accuracy and parameter interpretability. Nevertheless, some room remains for improvement of its prediction accuracy because it does not optimize the degree of forgetting of past data. Specifically, although its forgetting parameters are optimized solely using current input data, it should use both current input and past data to optimize them. Therefore, for better parameter estimation to improve accuracy, this study proposes a new DeepIRT that optimizes the degree of forgetting of past data. The proposed method has a hypernetwork to balance both the current and the past data in memory, which stores a student's knowledge states. Results of experiments demonstrate that the proposed method improves the prediction accuracy compared to earlier KT methods.

Keywords

Deep Learning, Hypernetwork, Item Response Theory, Knowledge Tracing

1. INTRODUCTION

Recently, with the development of online education [24, 25, 26], Knowledge Tracing (KT) has attracted broad attention for helping students to learn effectively by presenting optimal problems and a teacher's support. [3, 7, 10, 15, 16, 17, 28, 29, 33, 34, 35]. Important tasks of KT are tracing the student's evolving knowledge state and discovering concepts that the student has not mastered based on the student's past learning history data. Furthermore, accurate prediction of a student's performance (correct or incorrect response to an unknown item) is important for adaptive learning. Although KT methods have been proposed as probabilistic

approaches [3, 7, 28, 29, 34] and deep-learning-based approaches [17, 28, 29, 33, 35], the latter have been studied more actively in recent years because they reportedly have high prediction accuracies.

Various deep-learning-based approaches have been proposed to improve the prediction accuracy of a student's performance [1, 20, 21, 32]. Most recently, Ghosh et al. (2020) proposed attentive knowledge tracing (AKT) [5], which incorporates a forgetting function of past data to attention mechanisms: the Transformer method [27]. In addition, AKT optimizes the parameters to weight the data necessary for student performance prediction from past learning data. Therefore, AKT has the best performance for predicting a student's responses among earlier KT methods. However, the interpretability of the parameters is limited because it cannot express a student's ability transition of each skill [5, 14, 22].

On the other hand, to express a student's the knowledge state transition for deep-learning-based approaches, Zhang proposed the dynamic key-value memory network (DKVMN) [35]. DKVMN traces the knowledge state transition using a Memory-Augmented Neural Network and attention mechanisms. It can estimate the relations between underlying skills and items addressed by students. In addition, DKVMN has a memory updating component to allow forgetting and updating of the latent variable memory, which stores the students' knowledge states in the learning process [35]. For interpretability of the parameters, the memory updating component in DKVMN is more effective than the forgetting function of AKT because it updates the current latent variable memory, which stores the students' skills and abilities, using only the immediately preceding values.

To improve the interpretability of the parameters of DKVMN, DeepIRT was proposed by combining DKVMN with an Item Response Theory (IRT) [2, 11, 30] module [33]. It includes the students' ability parameters and the items' difficulty parameters. However, it was insufficient to improve the interpretability because a student's ability of DeepIRT depends on each item characteristic. To resolve this shortcoming, Tsutsumi et al. proposed DeepIRT methods with independent redundant student and item networks [22, 23]. They can learn the student's ability and item difficulty independently to avoid impairing the predictive accuracy. For DeepIRT [23], a student's ability is constant throughout a learning process because it is structured for test theory.

E. Tsutsumi, Y. Guo, and M. Ueno. DeepIRT with a hypernetwork to optimize the degree of forgetting of past data. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 543–548, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853107>

Therefore, it can not be applied to KT. To apply DeepIRT to KT, DeepIRT [22] was proposed using architecture of DKVMN. In DeepIRT [22], a student network employs memory network architecture to reflect dynamic changes of student abilities as DKVMN does. Because the student’s ability parameters of the DeepIRT [22] are independent of each item characteristic, it has higher interpretability than the earlier method has [33]. Furthermore, the DeepIRT [22] can express a student’s ability transition for each skill by estimating relations among the multidimensional skills. Consequently, the DeepIRT provides high interpretability without impairing the predictive accuracy.

However, room for improvement of prediction accuracy of the DeepIRT remains [22] because it does not optimize the degree of forgetting the past data. Specifically, in DKVMN and DeepIRT methods, the forgetting parameters which control the degree of forgetting the past data are optimized from only the current input data: the student’s latest response to an item. As a result, it might degrade the prediction accuracy of the DeepIRT because the value memory insufficiently reflects the past learning history data. Namely, it might be difficult to reflect the past data accurately in a long learning process. It should use not only the current input data but also past data to optimize the forgetting parameters.

In this study, we propose the new DeepIRT with a hypernetwork to optimize the forgetting parameters. The hypernetwork [4, 6, 8, 9, 12, 13, 19, 31] balances both current and the past data in the latent variable memory, which stores a student’s knowledge state data. Before the model updates the latent variable memory, it optimizes not only the weights of the forgetting parameters but also the past latent variable memory. Experiments were conducted to compare the performances of the proposed method and those of the earlier KT methods. The results demonstrate that the proposed method improves the prediction accuracy of the DeepIRT [22]. They also indicate the proposed method as effective, especially for tasks with a long-term learning process.

2. DKVMN AND DEEP-IRT METHODS

DKVMN and DeepIRT methods [22, 33, 35] have the same memory updating component to update and forget the students’ knowledge states in the learning process [35]. The value memory \mathbf{M}_t^v , which traces the process of student ability growth, is updated in this memory updating component. They use \mathbf{c}_j based on input \mathbf{q}_j , which reflects a latest student’s response data u_{tj} to item j at time t .

$$\mathbf{c}_j = \begin{cases} [\mathbf{0}, \mathbf{q}_j] & u_{tj} = 1 \\ [\mathbf{q}_j, \mathbf{0}] & u_{tj} = 0. \end{cases} \quad (1)$$

Here, $\mathbf{0}$ is a zero vector consisting of J zero values. They updated the value memory \mathbf{M}_t^v as

$$\mathbf{v}_t = \mathbf{W}^v \mathbf{c}_j + \boldsymbol{\tau}^v, \quad (2)$$

$$\mathbf{e}_t = \sigma(\mathbf{W}^e \mathbf{v}_t + \boldsymbol{\tau}^e), \quad (3)$$

$$\mathbf{a}_t = \tanh(\mathbf{W}^a \mathbf{v}_t + \boldsymbol{\tau}^a), \quad (4)$$

$$(5)$$

and

$$\tilde{\mathbf{M}}_{t+1,l}^v = \mathbf{M}_{t,l}^v \otimes (1 - w_{tl} \mathbf{e}_t) + w_{tl} \mathbf{a}_t^\top, \quad (6)$$

where \mathbf{W}^v , \mathbf{W}^e and \mathbf{W}^a are the weight matrices, and $\boldsymbol{\tau}^v$, $\boldsymbol{\tau}^e$ and $\boldsymbol{\tau}^a$ are the bias vectors. Furthermore, w_{tl} signifies the degree of strength of the relations between the underlying skill l and skill tags addressed by a student at time t . It is noteworthy that \mathbf{e}_t , and \mathbf{a}_t are forgetting parameters, which adjust the degrees of forgetting the past data and reflecting the current input data. \mathbf{e}_t influences how much the value memory forgets (remembers) the past ability. Additionally, \mathbf{a}_t controls how much the value memory reflects the current input data.

For the interpretability of the parameters, this memory updating component is more effective than the forgetting function of AKT because it updates the current latent variable memory which stores the student’s skills and abilities using only the immediately preceding values. However, the forgetting parameters are optimized only from current input data. It should use not only the current input data but also past data to optimize them. Additionally, the weights are fixed values and are not optimized for each time point. As a result, DKVMN and DeepIRT might degrade the prediction accuracies because of value memory $\mathbf{M}_{t,l}^v$ which only insufficiently reflects past learning history data. Especially, it might be difficult to reflect past data accurately in a long learning process.

3. PROPOSED METHOD

The preceding section described that the forgetting parameters of DeepIRT are not optimized using both current input data and past data. However, when using both current input data and past data, it is difficult to optimize the weight parameters directly because the number of parameters increases dynamically.

Recent studies in the field of Natural Language Processing (NLP) proposed the extension components to LSTM [18] in the form of mutual gating of the current input data and the previous output hidden variables [6]. These extension components are called hypernetworks. A hypernetwork supports the main recurrent neural network by optimizing the non-shared weights for each time point in the hidden layers [6]. In standard LSTM [18], the hidden variables change with time, but the weights used to update them are fixed values and are not optimized for each time point. To resolve this difficulty, various hypernetworks have been proposed to optimize the non-shared weights in the LSTM at each time point. [4, 6, 8, 9, 12, 13, 31]. Their results demonstrate that LSTM with a hypernetwork works better than the standard LSTM [18].

Melis et al. earlier proposed the ”Mogrifier component” which is a kind of hypernetwork for LSTM in the field of NLP [12]. Mogrifier also scales the weights and the hidden variables using not only the current inputs but also the output of the hidden variable at the previous point in time. They reported that the LSTM with Mogrifier component outperforms the other methods for a long input data length. Inspired by those studies, this study proposes a new hypernetwork that optimizes the degree of forgetting of past data in the DeepIRT [22] to improve prediction accuracy with the parameter interpretability. We incorporate the proposed hypernetwork in the memory updating component, which updates the latent variable \mathbf{M}_t^v , to avoid greatly increasing

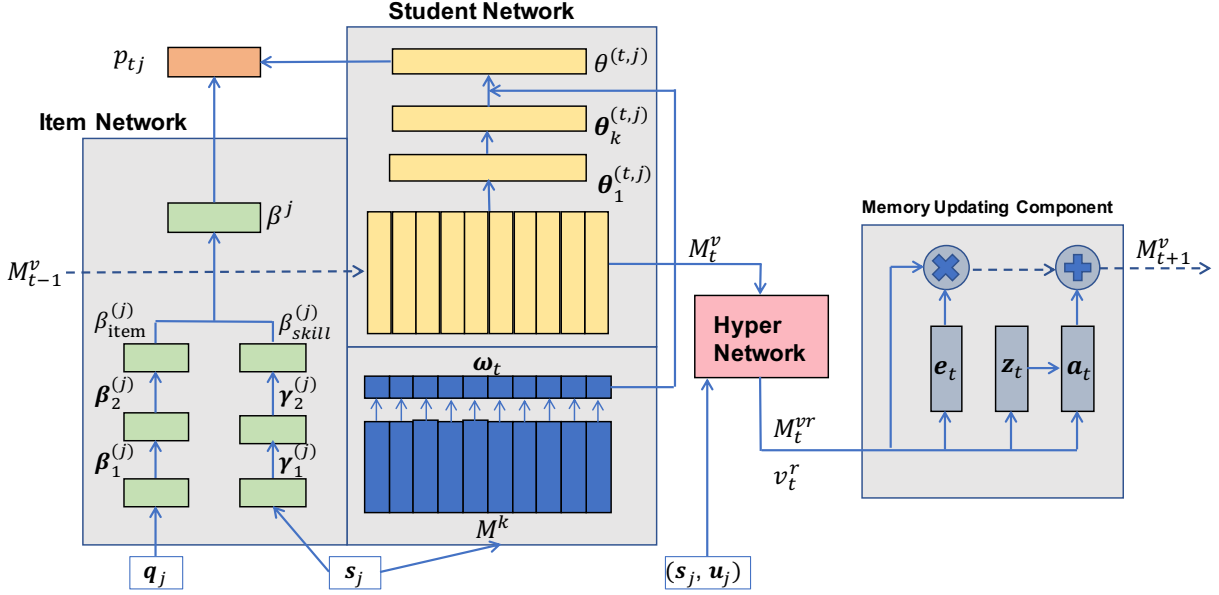


Figure 1: Network architecture of the proposed DeepIRT.

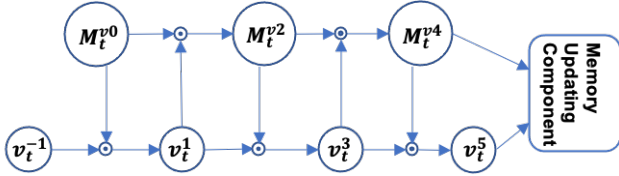


Figure 2: Proposed hypernetwork architecture.

number of parameters. Before the model updates the latent variable M_{t+1}^v , the proposed hypernetwork optimizes not only the weights of the forgetting parameters but also the past latent variable M_t^v . The proposed hypernetwork estimates the optimal forgetting parameters by balancing both the current input data and the past latent variable. In addition, the Mogrifier component [12] used constant values as the tuning parameters in the hypernetwork. For this study, we optimize the tuning parameters to adjust the hypernetwork for each dataset. No report of the relevant literature has described a study of the use of the hypernetworks for KT methods. Figure 1 presents the architecture of the proposed method. The right side of Figure 1 presents the hypernetworks and the memory updating component. The left side of Figure 1 shows the independent student and item networks.

3.1 Hypernetwork

To optimize the forgetting parameters at time t , the proposed hypernetwork balances the current input data and the past value memory M_t^v to store sufficient information of the learning history data before calculating the latent variables M_{t+1}^v . The proposed hypernetwork structure is located at the beginning of the Memory Updating Component on the right side of Figure 1 (shown in red).

Figure 2 shows the structure of the proposed hypernetwork. The inputs of the hypernetwork are the past value memory

M_t^v and current input data $(s_j, u_j) = s_j + u_j * S$ when a student responds to item j of skill s_j . Therein, $S \in \{1, 2, \dots, 2S\}$ represents the number of skills. The embedding vector of (s_j, u_j) denoted as $v_t \in \mathbb{R}^{d_v}$. Because of the repeating multiplications as shown in Figure 2, this hypernetwork balances current data v_t and past value memory M_t^v . For the proposed methods, we optimize the number of rounds r for each learning dataset.

3.2 Memory Updating Component

Next, we estimate the forgetting parameters e_t and a_t using the optimized v_t^r and M_t^{vr} in the hypernetwork. These forgetting parameters are important to update the latest value memory M_{t+1}^v optimally. The earlier memory updating component of DKVMN and DeepIRT methods calculates the forgetting parameters from v_t with only current input information in equation (3), (4). By contrast, we calculate them using the optimized current input data v_t^r and the past latent value M_t^{vr} . Therefore, the forgetting parameters e_t , and a_t are also estimated as optimizing the degree of forgetting of past data and as reflecting the current input data. Furthermore, the proposed method can capture the student knowledge state changes accurately because the latent knowledge state M_t^v has sufficient information of the past learning history data.

4. EXPERIMENTATION

4.1 Datasets and Experiment Setting

This section presents comparisons of the prediction accuracies of the proposed method with those of earlier methods (Tsutsumi et al. and AKT) [5, 22]. We use two standard benchmark datasets ASSISTments2009 and ASSISTments2017 collected from an online tutoring system. Table 1 presents the number of students (No. Students), the number of skills (No. Skills), the number of items (No. Items), the rate of correct responses (Rate Correct), and the aver-

Table 1: Summary of datasets

Dataset	No. students	No. skills	No. Items	Rate Correct	Learning length
ASSISTments2009	4151	111	26684	63.6%	52.1
ASSISTments2017	1709	102	3162	39.0%	551.0

Table 2: Prediction accuracies of students’ performances

Dataset	metrics	Tsutsumi et al.	AKT	Proposed
ASSISTments2009	AUC	80.70 +/- 0.56	82.20 +/- 0.25	81.57 +/- 0.39
	Acc	76.13 +/- 0.58	77.30 +/- 0.55	76.85 +/- 0.56
	Loss	0.54 +/- 0.10	0.49 +/- 0.10	0.53 +/- 0.13
ASSISTments2017	AUC	74.15 +/- 0.27	74.54 +/- 0.21	76.85 +/- 0.39
	Acc	68.73 +/- 0.11	69.83 +/- 0.15	71.08 +/- 0.50
	Loss	0.57 +/- 0.06	0.58 +/- 0.06	0.55 +/- 0.06
Average	AUC	77.42	78.37	79.21
	Acc	72.43	73.56	74.00
	Loss	0.56	0.54	0.54

age length of the items which students addressed (Learning length).

We used five-fold cross-validation to evaluate the prediction accuracies of the methods. The item parameters and hyper-parameters are trained by 70% of each dataset. Given the estimated parameters, the students’ abilities are estimated at each time using the remaining 30% of each dataset according to an earlier study [22]. We employ Adam optimization with a learning rate of 0.003 and batch-size 32. Additionally, 200 items was set as the upper limit of the input length according to the earlier studies [22, 33, 35]. For this study, we leverage three metrics for prediction accuracy: Accuracy (Acc) score, AUC score, and Loss score.

4.2 Prediction Accuracy

The respective values of Acc, AUC, and Loss for ASSISTments2009 and ASSISTments2017 datasets [5, 22] are presented in Table 2. We compared the performances of the proposed method with those of DeepIRT [22] and AKT for each dataset with item and skill tag inputs according to [5]. Additionally, this report describes the standard deviations across five test folds.

Results indicate that the proposed method, which optimizes the forgetting parameters, provides the best average scores for all metrics. Especially, the proposed method outperforms the Tsutsumi et al. [22] and AKT for ASSISTments2017. ASSISTments2017 has a long learning length. By contrast, the proposed method tends to have lower prediction accuracies for ASSISTments2009 with a shorter learning length than AKT has. Results suggest that the proposed hypernetwork functions effectively, especially for datasets with long learning lengths.

5. CONCLUSIONS

Recently, to express a student’s the knowledge state transition for deep-learning-based approaches, DKVMN and DeepIRT methods have been proposed. Tsutsumi et al. (2021) proposed a DeepIRT with independent redundant student and item networks [22]. It can learn the student’s ability and

item difficulty independently to avoid impairing the predictive accuracy. Furthermore, the DeepIRT [22] can express a student’s ability transition for each skill by estimating relations among the multidimensional skills. the DeepIRT [22] has a memory updating component to allow forgetting and updating of the latent variable memory, which stores the students’ knowledge states in the learning process. However, the forgetting parameters which control the degree of forgetting the past data are optimized from only the current input data. It might degrade the prediction accuracy of the DeepIRT because the value memory insufficiently reflects the past learning history data. It should use not only the current input data but also past data to optimize the forgetting parameters.

This study proposed a new DeepIRT with a hypernetwork that optimizes the degree of forgetting of the past data for parameter estimation to improve prediction accuracy with the parameter interpretability. In the proposed method, the hypernetwork balances the current input data and the past value memory to store sufficient information of the learning history data before calculating the latent variables. Specifically, it scales not only the weights of the forgetting parameters but also the hidden variables using the current inputs and the output of the hidden variable at the previous point in time.

Experiments conducted with the benchmark datasets demonstrated that the proposed method improves the prediction accuracies of the earlier KT methods. Especially, results showed that the proposed method is effective for tasks with a long-term learning process. As future work, we will evaluate the interpretability of the ability parameters of the proposed method by comparing the parameter estimates with those of the earlier DeepIRTs [22, 33]. Furthermore, we will clarify the mechanism of how the proposed hypernetwork functions to increase the predictive accuracy.

6. ACKNOWLEDGMENTS

This research was supported by JSPS KAKENHI Grant Numbers JP19H05663, JP19K21751 and JP22J15279.

7. REFERENCES

- [1] F. Ai, Y. Chen, Y. Guo, Y. Zhao, Z. Wang, G. Fu, and G. Wang. Concept-aware deep knowledge tracing and exercise recommendation in an online learning system. In *EDM*, 2019.
- [2] F. Baker and S. Kim. *Item Response Theory: Parameter Estimation Techniques, Second Edition*. Statistics: A Series of Textbooks and Monographs. Taylor & Francis, 2004.
- [3] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Model. User-Adapt. Interact.*, 4(4):253–278, Dec 1995.
- [4] C. Fernando, D. Banarse, M. Reynolds, F. Besse, D. Pfau, M. Jaderberg, M. Lanctot, and D. Wierstra. Convolution by evolution: Differentiable pattern producing networks. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pages 109–116, 07 2016.
- [5] A. Ghosh, N. Heffernan, and A. S. Lan. Context-aware attentive knowledge tracing. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020.
- [6] D. Ha, A. Dai, and Q. V. Le. Hypernetworks. *arXiv preprint arXiv:1609.09106*, 2016.
- [7] M. Khajah, Y. Huang, J. Gonzalez-Brenes, M. Mozer, and P. Brusilovsky. Integrating knowledge tracing and item response theory: A tale of two frameworks. *Personalization Approaches in Learning Environments*, 1181:5–17, 2014.
- [8] J. Koutník, F. Gomez, and J. Schmidhuber. Evolving neural networks in compressed weight space. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pages 619–626, 01 2010.
- [9] B. Krause, L. Lu, I. Murray, and S. Renals. Multiplicative lstm for sequence modelling. *Workshop Track in ICLR*, 2017.
- [10] J. Lee and E. Brunskill. The impact on individualizing student models on necessary practice opportunities. In *Proceedings of the Fifth International Conference on Educational Data Mining*, pages 118–125, 01 2012.
- [11] F. Lord and M. Novick. *Statistical Theories of Mental Test Scores*. Addison-Wesley, 1968.
- [12] G. Melis, K. Tomáš, and B. Phil. Mogrifier lstm. In *Proceedings of ICLR 2020*, 2020.
- [13] M. Moczulski, M. Denil, J. Appleyard, and N. Freitas. Acdc: A structured efficient linear layer. In *ICLR*, 2016.
- [14] S. Pandey and G. Karypis. A self-attentive model for knowledge tracing. In *Proceedings of International Conference on Education Data Mining*, 2019.
- [15] Z. Pardos and N. Heffernan. T.: Modeling individualization in a bayesian networks implementation of knowledge tracing. In *Proceedings of the 18th International Conference on User Modeling, Adaption, and Personalization*, pages 255–266, 06 2010.
- [16] Z. Pardos and N. Heffernan. Kt-idem: Introducing item difficulty to the knowledge tracing model. In *Proceedings of 19th International Conference on User Modeling, Adaptation and Personalization (UMAP 2011)*, pages 243–254, 01 2011.
- [17] C. Piech, J. Bassen, J. Huang, S. Ganguli, M. Sahami, L. J. Guibas, and J. Sohl-Dickstein. Deep knowledge tracing. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 505–513. Curran Associates, Inc., 2015.
- [18] H. Sepp and S. Jurgen. Long short-term memory. *Neural Computation*, 14:1735–1780, 1997.
- [19] K. Stanley, D. D’Ambrosio, and J. Gauci. A hypercube-based encoding for evolving large-scale neural networks. *Artificial life*, 15:185–212, 02 2009.
- [20] Y. Su, Q. Liu, Q. Liu, Z. Huang, Y. Yin, E. Chen, C. H. Q. Ding, S. Wei, and G. Hu. Exercise-enhanced sequential modeling for student performance prediction. In *AAAI*, pages 2435–2443, 2018.
- [21] X. Sun, X. Zhao, Y. Ma, X. Yuan, F. He, and J. Feng. Multi-behavior features based knowledge tracking using decision tree improved dkvmm. In *Proceedings of the ACM Turing Celebration Conference – China*, New York, NY, USA, 2019. Association for Computing Machinery.
- [22] E. Tsutsumi, R. Kinoshita, and M. Ueno. Deep-irt with independent student and item networks. In *Proceedings of the 14th International Conference on Educational Data Mining (EDM)*, 2021.
- [23] E. Tsutsumi, R. Kinoshita, and M. Ueno. Deep item response theory as a novel test theory based on deep learning. *Electronics*, 10(9), 2021.
- [24] M. Ueno. Data mining and text mining technologies for collaborative learning in an ILMS “samurai”. In *Proceedings of the IEEE International Conference on Advanced Learning Technologies, ICALT 2004*, pages 1052–1053, 2004.
- [25] M. Ueno and Y. Miyazawa. Probability based scaffolding system with fading. In *Proceedings of Artificial Intelligence in Education – 17th International Conference, AIED*, pages 237–246, 2015.
- [26] M. Ueno and Y. Miyazawa. IRT-based adaptive hints to scaffold learning in programming. *IEEE Transactions on Learning Technologies*, 11:415–428, 10 2018.
- [27] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. pages 5998–6008, 2017.
- [28] X. Wang, J. Berger, and D. Burdick. Bayesian analysis of dynamic item response models in educational testing. *The Annals of Applied Statistics*, 7(1):126–153, 2013.
- [29] R. Weng and D. Coad. Real-time bayesian parameter estimation for item response models. *Bayesian Analysis*, 13, 12 2016.
- [30] W.J. van der Linden. *Handbook of Item Response Theory, Volume Two: Statistical Tools*. Chapman and Hall/ CRC Statistics in the Social and Behavioral Sciences. Chapman and Hall/ CRC, 2016.
- [31] Y. Wu, S. Zhang, Y. Zhang, Y. Bengio, and R. Salakhutdinov. On multiplicative integration with recurrent neural networks. In *Advances in neural information processing systems*, pages 2856–2864, 2016.

- [32] X. Xiong, S. Zhao, V. Inwegen, E. G., and J. E. Beck. Going deeper with deep knowledge tracing. In *Proceedings of International Conference on Educational Data Mining*, 2016.
- [33] C. Yeung. Deep-irt: Make deep learning based knowledge tracing explainable using item response theory. In *Proceedings of the 12th International Conference on Educational Data Mining, EDM*, 2019.
- [34] M. V. Yudelson, K. R. Koedinger, and G. J. Gordon. Individualized bayesian knowledge tracing models. In *Artificial Intelligence in Education*, pages 171–180, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.
- [35] J. Zhang, X. Shi, I. King, and D.-Y. Yeung. Dynamic key-value memory network for knowledge tracing. In *Proceedings of the 26th International Conference on World Wide Web, WWW '17*, pages 765–774, Republic and Canton of Geneva, CHE, 2017. International World Wide Web Conferences Steering Committee.

Mining and Assessing Anomalies in Students' Online Learning Activities with Self-supervised Machine Learning

Lan Jiang
University of Illinois Urbana–Champaign
Champaign, IL, USA
lanj3@illinois.edu

Nigel Bosch
University of Illinois Urbana–Champaign
Champaign, IL, USA
pnb@illinois.edu

ABSTRACT

Two students in the same course working toward the same learning objectives may have very different strategies. However, on average, there are likely to be some patterns of student actions that are more common than others, especially when students are implementing typical self-regulated learning strategies. In this paper, we focus on distinguishing between students' typical actions and unusual, anomalous sequences of actions. We define anomalous activities as unexpected activities given a student's preceding activities. We distinguish these anomalies by training a self-supervised neural network to determine how predictable activities happen (the complement of which are anomalies). A random forest model trained to predict course grades from anomaly-based features showed that anomalous actions were significant predictors of course grade (mean Pearson's $r = .399$ across 7 courses). We also explore whether humans regard the anomalous activities labeled by the model as anomalies by asking people to label 20 example sequences. We further discuss the implications of our method and how detecting and understanding anomalies could potentially help improve students' learning experiences.

Keywords

Anomalies, Human understanding of anomalies, Log activities

1. INTRODUCTION

Online education systems can provide personalized learning experiences by understanding students' learning behavior automatically, given rich data that can be collected through such systems [18, 6, 12, 33]. Most research in this area focuses on investigating specific, theory-driven phenomena via data analytics and employs data-driven approaches to understand typical learning behaviors (the most frequent actions, most commonly studied resources, etc.) [33, 44, 23] and prediction tasks (grade/dropout prediction, test recommendation, etc.) [6, 25, 32, 41, 35]. These approaches, while

valuable, rarely consider the role of *anomalous* behaviors, which are also important to understand. For example, existing work has shown that some anomalous behaviors positively correlate with high course grade [17]. Much remains to be discovered regarding anomalous actions, how to determine which kinds of activities are anomalous activities, and how humans perceive anomalies. In this paper, we define students' activities in terms of typical (i.e., predictable) activities and anomalous activities (i.e., unpredictable activities), and describe a method for uncovering these anomalies. In addition, we investigate whether human experts' perceptions of anomalies align with anomalies identified by the proposed method, and explore how humans distinguish anomalous versus typical student activities.

We focus on data from log files [5], which accumulate a great deal of interaction information to understand anomalies in student learning behaviors. Due to the large amount of data in log files, it is difficult to glean insights from these manually. Thus, researchers have devised methods like behavior mining to extract insights computationally [20, 31]. However, behavior-based inferences mainly rely on handcrafted features [40, 43, 36] (e.g., number of occurrences of specific activities), which usually capture frequent or expected activities. Conversely, there may be anomalous activities that relate to learning as well, which are—by definition—unexpected and thus difficult to discover. In this paper, we propose and evaluate a generalizable approach to reveal anomalous activities by examining the prediction errors of neural networks.

Analyzing anomalies requires defining them, which may be difficult. Manual examination and inference based on expert knowledge is one possible approach for discovering specific constructs in data [30]. However, defining and determining anomalies is time-consuming and constrained to the limits of expert knowledge. In statistics and data mining, anomalous activities refer to data deviating from patterns exhibited by the majority of data [13, 29]. In this perspective, anomalous activities are those where, given a sequence of activities, the activities that followed are unexpected—similar to definitions for time series data [26, 42]. Anomalies in this definition indicate deviations from predictable learning strategies [14], and thus may be the result of deviations from common learning strategies or from the ways in which instructors expect students to go through course materials. Consequently, a method to discover students' anomalous learning behaviors might inspire changes to our understanding

L. Jiang and N. Bosch. Mining and assessing anomalies in students' online learning activities with self-supervised machine learning. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 549–554, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852948>

of e-learning strategies and could eventually help refine the design of learning experiences.

We approach this problem by training a self-supervised neural network to learn typical activity sequences, then detect anomalies based on the prediction errors. We demonstrate one aspect of the usefulness of our method by exploring the correlation between students' anomalous actions and students' learning outcomes. We further contribute to this problem by understanding how humans perceive anomalous activities and whether those perceptions are aligned with proposed approach.

2. RELATED WORK

In this section, we first discuss the concept of anomalies and provide an overview of applications of anomalies to highlight the potential for work in this area with educational data (section 2.1). We then investigate existing research on log data from learning management systems (section 2.2) to show the importance of behavioral data and how our approach contributes to related work in this area.

2.1 Anomalies

Anomalies are generally understood as rare data that do not conform to preconceptions or expectations derived from the majority of data [7]. Anomalies can be identified with statistical and machine learning techniques, and in various types of data, such as images and time series data [34, 3, 37, 26, 42]. However, in the context of students' behavioral sequences, anomalies are relatively poorly understood.

In image and video data (outside of educational contexts), anomalies refer to a set of features that are not expected, which provides context for how anomalies are defined and detected in general. For time series data, the data are linearly ordered and the definition of anomalies may differ as a result. A particular data value could be an anomaly in a specific context, and might be considered typical (not anomalous) in other contexts. Malhotra et al. [26] and Zhang et al. [42] leveraged prediction errors as an indication of anomalies.

We are aware of only one study that focused on anomalies in education-related sequential data [37]. They considered response time as an indicator of anomalous learning. After plotting the sequence of response times, the authors derived a posterior predictive distribution and regarded the learners as anomalous when they had an unusually high or low response time. However, time spent is not the only way in which actions might be anomalous; moreover, unusually high or low response times might actually be expected for some students when considered in the context of their previous behaviors.

An alternative way to distinguish anomaly versus normal actions is to analyze them in the context of a student's sequence of behaviors, which is the approach we take in this paper.

2.2 Data Mining in Log Activities

In recent years, there has been an increasing interest in analyzing log activities from e-learning environments. Re-

searchers have done a large number of tasks that try to understand students' behaviors, academic performance, and learning processes [10, 1, 28, 38, 39].

Much of the work [9, 44] on data-driven discovery in education focuses on extracting frequent sequential patterns that are common and thus may characterize the behaviors of students from a specific group or across an entire dataset. In contrast, our focus is on behaviors that distinguish students from their peers. Some existing works [30, 15, 19] build behavior models that incorporate relationships between past activities and current activities, or past state and current state, as we do in this study. Other works that rely on log files have explored connections between students' actions and high-level learning information. One research direction is to detect students' learning behaviors or learning preferences [25, 32, 41] as evident in logged activity data, with the goal of enabling personalization of learning experiences after identifying students' needs and preferences.

The methods discussed above do not directly examine anomalies; they rely on extracting features from log files to discover connections between those features and student outcomes or states, or to explore properties of the learning domain and task itself. However, these studies do show that the behavioral data reflects various states of students, and is thus a promising area for further exploration, such as with respect to anomalous behaviors.

3. METHOD

In this section, we describe the data used in our study and present the methods used to answer each of the research questions stated in the Introduction.

3.1 Dataset

We analyzed the Open University Learning Analytics Dataset (OULAD) [24] in our experiments. The data included in OULAD were collected from 2013 to 2014. The dataset contains information about 22 sections of 7 different courses (labeled *A* through *G*), including 32,593 students and their aggregated interactions with an LMS in terms of per-day counts of different types of actions.

We combined multiple sections of the same courses assuming that different sections of each course should be relatively similar. Based on the frequency of each activity, we observed that some activities rarely happened and appeared to be less meaningful for understanding student behaviors. In this work, we aim to detect anomalous activities in a given context instead of mining activities that happen rarely. We decided to group extremely uncommon interaction activities to simplify analysis and interpretation, though exploring the extremely rare events is one possible area for future work. Thus, we chose a threshold (i.e., less than twice per student on average) and grouped all interaction activities less frequent than the threshold into an "other" category. More information regarding specific actions can be found in existing work with this dataset (e.g., Figures 2 and 4 in [24], Table 2 in [21]).

3.2 Anomaly Detection

As discussed above, an "anomaly", generally speaking, refers to an unusual event. In our task, we formalized "typical"

events as predictable activities given a previous action sequence. In contrast, anomalies are students' activities that do not conform with predictions. Anomaly detection included two steps: (1) use machine learning to model typical activities and (2) measure the model's prediction errors. In particular, we trained a self-supervised sequential neural network to model activity sequences, leaving poorly-predicted activities as anomalies.

Our prediction model consists of three layers (though in principle the model could be expanded for datasets with more complex inputs): (1) the encoding layer, which is used for representation generation; (2) a sequential layer (e.g., convolution, recurrent), which is used for feature extraction; and (3) a fully connected layer with sigmoid activation to predict the next step in the sequence. In our experiments, we split the dataset into train and test sets with a ratio of 9:1. We ran experiments on two models: one with a convolutional layer for feature extraction, as described above, and an alternative model based on long short-term memory (LSTM) instead. The model takes three sequential actions, predicts the following action, and convolves over time. We set the kernel width of the convolutional layer to 3 and the number of filters to 20. For the LSTM model, we used a similar configuration (i.e., 20 LSTM cells). We trained models for 50 epochs with batch size 32. We used Adam as the optimizer [22] with a .004 learning rate for all seven courses after tuning the rate from .001 to .01 on course *A*.

To compute prediction error, we calculated the difference between actual actions and predicted actions for each timestamp. For each student, we computed the L2 loss (mean squared difference) between predicted actions and actual actions of each timestamp in the test set as an indicator of how well at each point a student conformed to expected behaviors. Thus, for a student activity sequence of length l , the error between actual and predicted action sequences can be represented by an l -length sequence, which has d dimensions (one for each action type).

3.3 Correlation with Grade

A common approach for predicting students' outcomes is to engineer features from students' activities that are believed to have some relationship to outcomes [4]. Analogously, we expected that a student's activity typicality (anomalous versus typical actions) directly relates to their learning outcomes if anomalous behaviors are evidence of adapting behaviors (e.g., via self-evaluation) or the opposite. We tested this hypothesis by calculating correlations between anomaly loss features and students' outcomes.

To model the relationship between activity typicality and student's outcomes, we represented each student with a d -dimensional vector where each element in the vector is the aggregated error for a specific action. That is, we defined *anomaly loss* as the aggregated error for each possible action for each student. We computed anomaly loss by aggregating error from anomaly detection at the student level by calculating the mean for each possible action. We further determined which actions were most important by training random forest regression with 25 trees to predict students' outcomes from the anomaly loss. During whole process, we conducted experiments only on the test set in the dataset

we used to model students' activities. We further split the test set into train and test subsets randomly in a 2:1 ratio, ensuring that data from each student appeared in only one subset or the other. The usefulness of anomaly loss for each action can then be calculated from the feature importance values of the random forest regressor.

3.4 Human Perceptions of Anomalies

Anomaly detection is mostly rooted in the statistics and machine learning communities [2, 7, 14]. Whether or not the anomalies detected by the proposed approach align with human intuition is still unclear. Thus, we conducted a small survey in which we asked four people with data mining experience to make their own assessments of anomalous student behaviors. We selected 10 sequences that were representative from the top 5% of most anomalous sequences, and 10 sequences from the 5% most typical. We presented 20 sequences (half typical activities, half anomalous activities), descriptions of the types of activities, and asked the participants (i.e., "coders") to determine whether the activities that happened on day 4 were typical or anomalous given three previous days of activities. After they finished coding the activities, we asked participants to provide insight into their coding strategy and their perceptions of what an anomaly is. Specifically, we asked them to describe how they perceive anomalies versus typical activities.

4. RESULTS AND DISCUSSION

4.1 Behavioral Prediction Models

The losses of CNN and LSTM models indicated that they worked approximately equally (mean of loss of LSTM was 1.3% higher; details can be found in the Appendix) well for feature extraction and the behavioral prediction task. We focused on the CNN model alone in the remainder of analyses for the sake of simplicity, since the two models were similarly accurate.

4.2 Correlation with Grade

In this section, we present the results of the analysis comparing anomalies and grade.

For all courses, predicted grade—from a random forest model with anomaly-based features—had a substantial correlation (mean $r = .399$, $SD = .073$) with actual grade. Correlations ranged from $r = .295$ (for course *D*) to $.494$ (for course *G*) with all p -values less than .05, indicating that correlations were positive across courses; even the lowest correlation indicated a moderate relationship between course grades and predictions made based on anomalies. Thus, we conclude that anomalies are important to investigate since they relate to students' academic outcomes.

To further explore which types of anomalous actions were most related to students' course grades, we analyzed feature importance of the random forest model. We used course *A* as an example. Feature importances in Table 1 show that the top five most important actions were *exam*, *other*, *oucontent*, *resource*, and *gap*. These actions were not necessarily the most common, yet still important because of their role in learning. Anomalous exam-related actions, in particular, explained over half of the model's feature importance (which sums to 1 in a random forest). Other important anomalous

actions may relate to self-regulated learning behavior; for example, *gap*-related anomalies may indicate irregular course participation.

Table 1: Importance for each feature in random forest grade prediction. Top five important actions shown in bold. Detailed information of actions can be found in existing work with this dataset (e.g., Figure 2 and 4 in [24], Table 2 in [21])

Action	Feature Importance	Action	Feature Importance
Exam	.596	Quiz	.001
Forumng	.020	Register	.006
Gap	.064	Resource	.054
Homepage	.018	Subpage	.008
Other	.085	Transfer	.012
Oucontent	.073	Unregister	.022
Ouwiki	.000	Url	.040

4.3 Results of Anomaly Detection

Self-supervised neural networks, such as the model we trained in this study, learn the conditional probability distribution of possible elements in a sequence [16, 27, 11]. In our anomaly detection framework in particular, the model learns the probability of each activity occurring given activities in the preceding three days. We provided example sequences of activities labeled by the proposed method in Appendix Table 4 in appendix. The criteria the model learns for predicting activities may be complex, but Appendix Table 4 does illustrate some reasonable high-level patterns learned by the model. For example, if the activities that occurred on day 4 were not consistent with preceding activities (either students performed many more or fewer than the previous activity pattern), they were tagged as anomalous. In contrast, if the activities on day 4 appeared in the previous days once or more, then they were usually labeled as typical. The conditional probability of an activity is just one way of defining anomalies, however, and may not align with human perceptions of what anomalies are. Thus, we also explored human perceptions of anomalies, as described next.

4.4 Results of Human Perception

For each pair of four human coders, we calculated Cohen’s kappa coefficient to measure their inter-rater agreement [8], as shown in Table 2.

Table 2: Kappa coefficients among machine learning and human coders of anomalous vs. typical sequences.

	Model	Coder 1	Coder 2	Coder 3
Coder 1	.00			
Coder 2	.60	.00		
Coder 3	.00	-.20	-.20	
Coder 4	.40	.20	.42	-.20

Inter-rater agreement results show that coder 2 and coder 4 agreed somewhat with each other and with the model: the kappa coefficient between coder 2 and the model was .60, between coder 4 and model it was .40, and between coder 2 and coder 4 kappa was .42. Conversely, both coder 1 and coder 3 had close to zero agreement with others: the mean kappa coefficients between coder 1, coder 3 and others

were .00 and -.13 respectively. Similarly, kappa coefficients among coder 1, coder 3, and the model were -.07, on average, indicating that coder 1 and coder 3 also did not agree with each other or with the model.

Coders’ perceptions of anomalies largely aligned with whether they agreed with each other and with the model: coder 2 and coder 4 determined anomalies from a more statistical perspective, while coder 1 and coder 3 mostly determined anomalies subjectively (they imagined whether or not the sequences were consistent with their own behaviors). In their descriptions of anomalies, they mentioned that if students had different activities (either fewer or much more activities) on day 4 than what happened on day 1 to day 3, they considered the activities on day 4 to be anomalous. For example, as coder 2 said:

Coder 2: The first [criterion I used] is if the activities included many more than what I expected to be there from the past three days. For example, if on days 1-3 the user only went to the Homepage plus one other place, and on day 4 the user went to many different places, I considered that anomalous.

Conversely, coder 1 suggested that if a student did not use the discussion forum on day 1 to day 3 but used it on day 4, then the activities on day 4 are anomalies. In addition, coder 1 thought if a student only watches content to prepare for an exam, then the exam happening on day 4 is an anomaly. Coder 3 decided the typicality of activities by linking them to his/her own experience. Coder 3 thought that classes rarely require a consistent effort on the same activities throughout any given week:

Coder 3: If activity types appear to be too consistent in Days 1-3, I became doubtful that any activity in Day 4 would be a typical activity. From my experience taking college courses, classes rarely require a consistent effort on the same types of activities throughout any given week, so too much consistency made me more likely to believe that the activities on Day 4 were anomalous.

5. CONCLUSIONS

Our goal in this study was to more deeply understand students’ behavior in web-based learning systems, specifically in terms of anomaly detection. We formally defined anomalies as unexpected activities given preceding activities and demonstrated that these anomalies are significantly related to student outcomes. We only tested our method with a selection of the previous three days’ activities as context for the next day’s activities prediction. However, larger fixed sequence intervals could be of interest. We further investigated if anomalies detected by our method aligned with human perceptions of anomalies, finding that the method indeed aligns with some conceptions of what anomalies are, though further research is needed to explore alternative conceptions. Ultimately, anomaly detection may lead to improvements in student modeling, activity recommendations, and even modifications of course materials and learning environments as researchers and teachers rely on methods like these to identify and address critical moments in learning processes.

6. REFERENCES

- [1] M. A. Abdullah. Learning style classification based on student's behavior in moodle learning management system. *Transactions on Machine Learning and Artificial Intelligence*, 3(1):28–40, 2015.
- [2] C. C. Aggarwal. An introduction to outlier analysis. pages 1–34, 2017.
- [3] S. Akcay, A. Atapour-Abarghouei, and T. P. Breckon. Ganomaly: Semi-supervised anomaly detection via adversarial training. In *Asian Conference on Computer Vision*, pages 622–637. Springer, 2018.
- [4] M. M. Ashenafi, G. Riccardi, and M. Ronchetti. Predicting students' final exam scores from their course activities. In *Frontiers in Education Conference (FIE)*, pages 1–9, 2015.
- [5] R. S. Baker and P. S. Inventado. *Educational data mining and learning analytics*. Springer, 2014.
- [6] W.-L. Chan and D.-Y. Yeung. Clickstream knowledge tracing: Modeling how students answer interactive online questions. In *Proceedings of the 11th International Learning Analytics and Knowledge Conference*, pages 99–109, 2021.
- [7] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):1–58, 2009.
- [8] J. Cohen. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20:37–46, 1960.
- [9] O. Dermay and A. Brun. Can we take advantage of time-interval pattern mining to model students activity? In *Proceedings of the 13th International Conference on Educational Data Mining*, pages 69–80. ERIC, 2020.
- [10] A. Dutt and M. A. Ismail. Can we predict student learning performance from lms data? a classification approach. In *Proceedings of the 3rd International Conference on Current Issues in Education*, pages 24–29. Atlantis Press, 2019.
- [11] S. R. Eddy. Hidden markov models. *Current opinion in structural biology*, 6(3):361–365, 1996.
- [12] E. Er, C. Villa-Torranó, Y. Dimitriadis, D. Gasevic, M. L. Bote-Lorenzo, J. I. Asensio-Pérez, E. Gómez-Sánchez, and A. Martínez Monés. Theory-based learning analytics to explore student engagement patterns in a peer review activity. In *Proceedings of the 11th International Learning Analytics and Knowledge Conference*, pages 196–206, 2021.
- [13] R. Foorthuis. A typology of data anomalies. In *International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, pages 26–38. Springer, 2018.
- [14] R. Foorthuis. On the nature and types of anomalies: A review of deviations in data. *International Journal of Data Science and Analytics*, 12(4):297–331, 2021.
- [15] C. Geigle and C. Zhai. Modeling mooc student behavior with two-layer hidden markov models. In *Proceedings of the 4th ACM Conference on Learning at Scale*, pages 205–208, 2017.
- [16] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber. Lstm: A search space odyssey. *IEEE transactions on neural networks and learning systems*, 28(10):2222–2232, 2016.
- [17] J. Huang, A. Dasgupta, A. Ghosh, J. Manning, and M. Sanders. Superposter behavior in mooc forums. In *Proceedings of the 1st ACM Conference on Learning at Scale*, pages 117–126, 2014.
- [18] Y. Huang, N. G. Lobczowski, J. E. Richey, E. A. McLaughlin, M. W. Asher, J. M. Harackiewicz, V. Aleven, and K. R. Koedinger. A general multi-method approach to data-driven redesign of tutoring systems. In *Proceedings of the 11th International Learning Analytics and Knowledge Conference*, pages 161–172, 2021.
- [19] P. Hur, N. Bosch, L. Paquette, and E. Mercier. Harbingers of collaboration? the role of early-class behaviors in predicting collaborative problem solving. *Proceedings of the 13th International Conference on Educational Data Mining*, pages 104–114, 2020.
- [20] A. Jalal and M. Mahmood. Students' behavior mining in e-learning environment using cognitive processes with information technologies. *Education and Information Technologies*, 24(5):2797–2821, 2019.
- [21] L. Jiang and N. Bosch. Predictive sequential pattern mining via interpretable convolutional neural networks. In *Proceedings of the 14th International Conference on Educational Data Mining*, pages 761–766. ERIC, 2021.
- [22] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *Proceeding of the 3rd International Conference on Learning Representations*, 2015.
- [23] J. S. Kinnebrew, K. M. Loretz, and G. Biswas. A contextualized, differential sequence mining method to derive students' learning behavior patterns. *Journal of Educational Data Mining*, 5(1):190–219, 2013.
- [24] J. Kuzilek, M. Hlosta, and Z. Zdrahal. Open university learning analytics dataset. *Scientific data*, 4(1):1–8, 2017.
- [25] M. P. P. Liyanage, L. G. KS, and M. Hirakawa. Detecting learning styles in learning management systems using data mining. *Journal of Information Processing*, 24(4):740–749, 2016.
- [26] P. Malhotra, L. Vig, G. Shroff, and P. Agarwal. Long short term memory networks for anomaly detection in time series. In *Proceeding of European Symposium on Artificial Neural Networks, Computational Intelligence, and Machine Learning*, volume 89, pages 89–94, 2015.
- [27] T. Mikolov, M. Karafiát, L. Burget, J. Cernocký, and S. Khudanpur. Recurrent neural network based language model. volume 2, pages 1045–1048. Makuhari, 2010.
- [28] B. Motz, J. Quick, N. Schroeder, J. Zook, and M. Gunkel. The validity and utility of activity logs as a measure of student engagement. In *Proceedings of the 9th International Learning Analytics and Knowledge Conference*, pages 300–309, 2019.
- [29] M. Nassir. Anomaly detection using principles of human perception. *arXiv preprint arXiv:2103.12323*, 2021.
- [30] L. Paquette and R. S. Baker. Comparing machine learning to knowledge engineering for student behavior modeling: a case study in gaming the system.

Interactive Learning Environments, 27(5-6):585–597, 2019.

[31] Y. Psaromiligkos, M. Orfanidou, C. Kytagiias, and E. Zafiri. Mining log data for the analysis of learners’ behaviour in web-based learning management systems. *Operational Research*, 11(2):187–200, 2011.

[32] S. Rajper, N. A. Shaikh, Z. A. Shaikh, and G. A. Mallah. Automatic detection of learning styles on learning management systems using data mining technique. *Indian Journal of Science and Technology*, 9(15):1–5, 2016.

[33] F. Rodriguez, H. R. Lee, T. Rutherford, C. Fischer, E. Potma, and M. Warschauer. Using clickstream data mining techniques to understand and support first-generation college students in an online chemistry course. In *Proceedings of the 11th International Learning Analytics and Knowledge Conference*, pages 313–322, 2021.

[34] W. Sultani, C. Chen, and M. Shah. Real-world anomaly detection in surveillance videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6479–6488, 2018.

[35] O. M. Teodorescu. Building test recommender systems for e-learning systems. In *Proceedings of the 13th International Conference on Educational Data Mining*, pages 810–814, 2020.

[36] N. Tomasevic, N. Gvozdenovic, and S. Vranes. An overview and comparison of supervised data mining techniques for student exam performance prediction. *Computers and Education*, 143:103676:1–18, 2020.

[37] M. Ueno. Online outlier detection system for learning time data in e-learning and it’s evaluation. *Proc. of Computers and Advanced Technology in Education (CATE2004)*, pages 248–253, 2004.

[38] S. Van Goidsenhoven, D. Bogdanova, G. Deeva, S. v. Broucke, J. De Weerd, and M. Snoeck. Predicting student success in a blended learning environment. In *Proceedings of the 10th International Conference on Learning Analytics and Knowledge*, pages 17–25, 2020.

[39] H. Wei, H. Li, M. Xia, Y. Wang, and H. Qu. Predicting student performance in interactive online question pools using mouse interaction features. In *Proceedings of the 10th International Conference on Learning Analytics and Knowledge*, pages 645–654, 2020.

[40] T.-Y. Yang, C. G. Brinton, C. Joe-Wong, and M. Chiang. Behavior-based grade prediction for moocs via time series neural networks. *IEEE Journal of Selected Topics in Signal Processing*, 11(5):716–728, 2017.

[41] F. Zehner, S. Harrison, B. Eichmann, T. Deribo, D. Bengs, N. Anderson, and C. Hahnel. The naep edm competition: On the value of theory-driven psychometrics and machine learning for predictions based on log data. In *Proceedings of the 13th International Conference on Educational Data Mining*, pages 302–312, 2020.

[42] C. Zhang, D. Song, Y. Chen, X. Feng, C. Lumezanu, W. Cheng, J. Ni, B. Zong, H. Chen, and N. V. Chawla. A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time

series data. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 1409–1416, 2019.

[43] W. Zhang, X. Huang, S. Wang, J. Shu, H. Liu, and H. Chen. Student performance prediction via online learning behavior analytics. In *2017 International Symposium on Educational Technology (ISET)*, pages 153–157. IEEE, 2017.

[44] Y. Zhang and L. Paquette. An effect-size-based temporal interestingness metric for sequential pattern mining. In *Proceedings of the 13th International Conference on Educational Data Mining*, pages 720–724, 2020.

APPENDIX

In the appendix, we provide details of our experiments. We include the losses of our behavior model for all courses in Appendix Table 3. We also introduce samples of anomalous activities and typical activities labeled by our method in Appendix Table 4.

Table 3: Loss (binary cross-entropy) of the behavioral prediction model based on different architectures.

Loss	Course A	B	C	D	E	F	G
LSTM	0.0039	0.0025	0.0028	0.0037	0.0035	0.0041	0.0026
CNN	0.0037	0.0024	0.0029	0.0037	0.0034	0.0042	0.0025

Table 4: Examples of activities labeled by our proposed approach The last column refers to the typicality of activities on day 4 given the activities from day 1 to day 3, which are labeled by the proposed approach.

Activities for each day				
Day 1	Day 2	Day 3	Day 4	Day 4 Type
Homepage	Forum	Forum	Exam	Anomalous
	Homepage	Homepage	Forum	
	Content		Homepage	
			Content	
			Resource	
			Subpage	
			URL	
Forum	Forum	Forum	Gap	Anomalous
Homepage	Homepage	Homepage		
Content	Content	Content		
Subpage	Subpage	Subpage		
URL	URL	URL		
Gap	Homepage	Gap	Exam	Anomalous
	Content			
	Subpage			
Forum	Homepage	Homepage	Homepage	Typical
Homepage				
Content				
Exam	Gap	Forum	Gap	Typical
		Homepage		
		Content		
		Resource		
		Subpage		
Homepage	Homepage	Homepage	Homepage	Typical
			Content	
			Subpage	
			URL	

Faster Confidence Intervals for Item Response Theory via an Approximate Likelihood Profile

Benjamin Paaßen
German Research Center for
Artificial Intelligence
Berlin, Germany
benjamin.paassen@dfki.de

Christina Göpfert
Faculty of Technology
Bielefeld University
Bielefeld, Germany

Niels Pinkwart
Institute of Computer Science
Humboldt-University of Berlin
Berlin, Germany

ABSTRACT

Item response theory models the probability of correct student responses based on two interacting parameters: student ability and item difficulty. Whenever we estimate student ability, students have a legitimate interest in knowing how certain the estimate is. Confidence intervals are a natural measure of uncertainty. Unfortunately, computing confidence intervals can be computationally demanding. In this paper, we show that confidence intervals can be expressed as the solution to a feature relevance optimization problem. We use this insight to develop a novel solver for confidence intervals and thus achieve speedups by 4-50x while retaining near-indistinguishable results to the state-of-the-art approach.

Keywords

item response theory, confidence intervals, relevance intervals, approximation

1. INTRODUCTION

Item response theory (IRT) is a well-established method to model the responses of students on a test [1, 5, 8]. The basic version models the probability of student i to answer correctly on item j as $p_{i,j} = 1/(1 + \exp[b_j - \theta_i])$, where θ_i is a parameter representing the ability of student i , and b_j is a parameter representing the difficulty of item j . If a student's ability exceeds the item's difficulty, the success probability is larger than 0.5, and vice versa. Numerous extensions have been developed over the years, such as parameters for item discrimination (two-parameter IRT), base success chance due to random guessing (three-parameter IRT) [1, 5], IRT for multiple skills [10], performance factor analysis, which describes the increased success chance for repeated attempts [12], or combinations with machine learning [11, 13].

Whenever we use IRT to estimate student ability, students have a legitimate interest in making sure that our model does

B. Paaßen, C. Göpfert, and N. Pinkwart. Faster confidence intervals for item response theory via an approximate likelihood profile. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 555–559, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852950>

not underestimate them (in line with the European Union's concept of a right to explanation [6, 9]). Accordingly, it is important that we not only estimate each student's ability, but also our uncertainty. We can quantify uncertainty in IRT models via confidence intervals [3, 4]. Roughly speaking, an α -confidence interval describes a range of possible parameter values such that the 'true' value is outside the range with probability at most α . For example, Wald's method assumes that parameters are Gaussian-distributed and uses the Gaussian's standard deviation to estimate confidence intervals [3]. While this is efficient, the Gaussian approximation assumes symmetry of the distribution and is only valid for large numbers of items in the test [4]. This is often unrealistic, which is why Wald's method tends to provide inaccurate confidence intervals in practice [4]. Doebler et al. have reacted by providing more exact methods under the assumption of constant item parameters but the resulting confidence intervals are disconnected regions [4], which are challenging to interpret.

More recently, Chalmers et al. [3] suggested the likelihood profile method, which is based on a likelihood ratio test. In particular, we look for a parameter range in which the log-likelihood is at least the optimal log-likelihood minus half the $1 - \alpha$ -quantile of the χ^2 -distribution with one degree of freedom. Because this method does not assume symmetry and works on the 'true' likelihood, it improves the accuracy of confidence intervals considerably. However, the computation requires a nested optimization scheme for every single parameter, which is computationally demanding, especially for large numbers of students.

In this paper, we provide a new perspective by showing that likelihood profile confidence intervals are equivalent to feature relevance intervals, which can be found via an optimization problem [7]. Second, we utilize this theoretical insight to develop a novel solver for confidence intervals which is considerably faster (by factors of 4-50x), while the resulting confidence intervals are almost indistinguishable from direct computation. We evaluate our proposed solver on a range of synthetic experimental conditions from 30-500 students and tests with 10 or 20 items. The experimental code can be found at https://github.com/bpaassen/ability_bounds.

2. METHOD

Our goal is to develop a faster solver for confidence bounds for ability parameters of IRT models via the likelihood profile technique [3]. For an illustration of the technique, con-

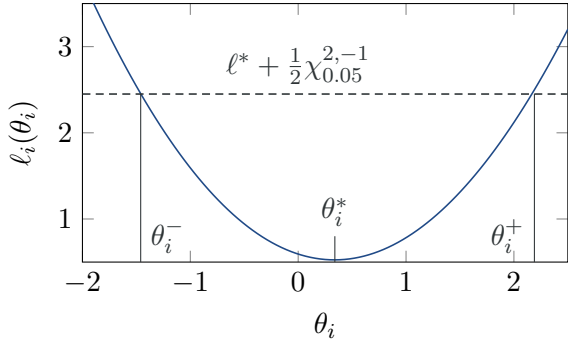


Figure 1: Illustration of the optimal NLL we can achieve for a certain value of θ_i in a single-student, single-item setup with $C = 1$ and $y_{1,1} = 1$. The dashed line shows the loss bound for $\alpha = .05$, the solid lines mark the corresponding confidence interval

sider Fig. 1. The blue curve illustrates the best negative log-likelihood (NLL) ℓ we can achieve when fixing the ability parameter θ_i to the value on the x axis. The likelihood profile technique now asks for the values θ_i^- and θ_i^+ , such that the NLL is exactly $\ell^* + \frac{1}{2}\chi_\alpha^{2,-1}$, where ℓ^* is the overall minimum NLL and $\chi_\alpha^{2,-1}$ is the $1 - \alpha$ quantile of the χ^2 distribution with one degree of freedom. Our α confidence interval is, then, given as $[\theta_i^-, \theta_i^+]$.

More precisely, the NLL of an IRT model on a dataset with m students and n items is given as

$$\begin{aligned} \ell(\vec{\theta}, \vec{b}) = & \sum_{i=1}^m \sum_{j=1}^n -y_{i,j} \cdot \log[p_{i,j}] - (1 - y_{i,j}) \cdot \log[1 - p_{i,j}] \\ & + \frac{1}{2C} \cdot (\|\vec{\theta}\|^2 + \|\vec{b}\|^2), \end{aligned} \quad (1)$$

where $y_{i,j}$ is 1 if student i answered correctly on item j and is 0, otherwise, $p_{i,j} = 1/(1 + \exp[b_j - \theta_i])$, and C is the variance of a Gaussian prior on the ability parameters $\vec{\theta}$ and item parameters \vec{b} [1] (chapter 7).

Now, let $\vec{\theta}_{-i}$ denote the vector $\vec{\theta}$ without its i th element and let $\ell_i(\theta_i) := \min_{\vec{\theta}_{-i}, \vec{b}} \ell(\vec{\theta}, \vec{b})$. In other words, $\ell_i(\theta_i)$ is the minimum NLL value we can achieve if we hold θ_i fixed but optimize all other parameters. Then, the likelihood profile method solves the equation $\ell_i(\theta_i) = \ell^* + \frac{1}{2}\chi_\alpha^{2,-1}$, where $\ell^* = \min_{\vec{\theta}, \vec{b}} \ell(\vec{\theta}, \vec{b})$. Because ℓ is convex in θ_i , this equation has exactly two solutions, which correspond to our interval bounds. The drawback of the likelihood profile method is its computational demand. For each student i , we need to solve a nested optimization problem, where the outer optimization searches for a solution to the equation $\ell_i(\theta_i) = \ell^* + \frac{1}{2}\chi_\alpha^{2,-1}$, and the inner optimization computes $\ell_i(\theta_i)$ for each value of θ_i probed by the outer optimization. We look for a way to speed up this computation.

Our key inspiration is the concept of feature relevance intervals (FRI) proposed by Göpfert et al. [7]. The FRI for some parameter θ is defined as the interval between the minimum and maximum value that still retains a loss of at most $(1 - \delta)$ times the optimal loss, where δ is a user-defined hy-

perparameter. More precisely, given some loss function ℓ of some parameter vector $\vec{\theta}$ and some specific parameter θ_i , the FRI for θ_i is computed by solving the following minimization/maximization problem:

$$\min_{\vec{\theta}} / \max_{\vec{\theta}} \theta_i \quad \text{s.t.} \quad \ell(\vec{\theta}) \leq \ell^* \cdot (1 + \delta). \quad (2)$$

Note that this concept is more general than confidence intervals and is mostly intended for classifiers in machine learning [7]. However, for IRT, confidence intervals via the likelihood profile method and FRIs happen to be equivalent.

THEOREM 1. *Let $\ell^* = \min_{\vec{\theta}, \vec{b}} \ell(\vec{\theta}, \vec{b})$ for the NLL (1) and some $C > 0$. Then, for any $\alpha \in (0, 1)$ and any $i \in \{1, \dots, m\}$, it holds: Problem (3) is convex with a global optimum $\vec{\theta}^\pm, \vec{b}^\pm$, such that $\ell(\vec{\theta}^\pm, \vec{b}^\pm) = \ell_i(\theta_i^\pm) = \ell^* + \frac{1}{2}\chi_\alpha^{2,-1}$.*

$$\min_{\vec{\theta}, \vec{b}} / \max_{\vec{\theta}, \vec{b}} \theta_i \quad \text{s.t.} \quad \ell(\vec{\theta}, \vec{b}) \leq \ell^* + \frac{1}{2}\chi_\alpha^{2,-1}. \quad (3)$$

PROOF. As a notational shorthand, let $L_\alpha := \ell^* + \frac{1}{2}\chi_\alpha^{2,-1}$. Note that the objective function of (3) is linear and hence convex. Now, consider the feasible set $\mathcal{X} = \{(\vec{\theta}, \vec{b}) | \ell(\vec{\theta}, \vec{b}) \leq L_\alpha\}$. Let $x, y \in \mathcal{X}$ and let $z_\gamma = \gamma \cdot x + (1 - \gamma) \cdot y$ for some $\gamma \in [0, 1]$, that is, z_γ is on the connecting line between x and y . Then, it must hold: $\ell(z_\gamma) \leq \gamma \cdot \ell(x) + (1 - \gamma) \cdot \ell(y)$, because ℓ is convex with respect to $\vec{\theta}$ and \vec{b} . Further, because $x, y \in \mathcal{X}$, $\ell(x) \leq L_\alpha$ and $\ell(y) \leq L_\alpha$. Therefore, $\ell(z_\gamma) \leq L_\alpha$, implying that $z_\gamma \in \mathcal{X}$. Hence, \mathcal{X} is convex and (3) is convex.

Next, note that, for any $\alpha \in (0, 1)$, $\chi_\alpha^{2,-1}$ is strictly larger zero. Therefore, the feasible set \mathcal{X} is not empty (it contains at least the minimizer of ℓ). In turn, problem (3) must have a optimum $(\vec{\theta}^\pm, \vec{b}^\pm)$ with $\ell(\vec{\theta}^\pm, \vec{b}^\pm) = L_\alpha$ because the objective function is strictly increasing.

Finally, we need to show that $\ell_i(\theta_i^\pm) = L_\alpha$. Assume this is not the case. If $\ell_i(\theta_i^\pm) > L_\alpha$, then $\min_{\vec{\theta}_{-i}, \vec{b}} \ell(\vec{\theta}, \vec{b}) > \ell(\vec{\theta}^\pm, \vec{b}^\pm)$, which is a contradiction. If $\ell_i(\theta_i^\pm) < L_\alpha$, we need to inspect the behavior of ℓ_i in more detail. Let (θ_i^*, b^*) be the solution to the unconstrained problem $\min_{\vec{\theta}, \vec{b}} \ell(\vec{\theta}, \vec{b})$. Then, we know that ℓ_i attains its minimum at θ_i^* with $\ell_i(\theta_i^*) = \ell^*$. Further, because ℓ_i is defined as the component-wise minimum of a convex function, it is convex itself [2] (p. 87). In turn, we know that for any $\theta_i > \theta_i^*$, ℓ_i is increasing and for any $\theta_i < \theta_i^*$, ℓ_i is decreasing. Because $C > 0$, it is also guaranteed that $\ell_i(\theta_i)$ exceeds L_α for sufficiently large/small θ_i , e.g., $\ell_i(\pm\sqrt{2 \cdot C \cdot L_\alpha}) > \frac{1}{2C} \sqrt{2 \cdot C \cdot L_\alpha}^2 = L_\alpha$. Now, consider the minimization version of (3). In that case, we certainly have $\theta_i^- \leq \theta_i^*$, otherwise θ_i^- would not be minimal. Now, because $\ell_i(\theta_i^-) < L_\alpha$, because ℓ_i is decreasing for all values $\theta_i < \theta_i^*$, and because ℓ_i is continuous, there must exist some value $\theta_i < \theta_i^-$ with $\ell_i(\theta_i^-) < \ell_i(\theta_i) \leq L_\alpha$. Therefore, $(\vec{\theta}^-, \vec{b}^-)$ is not a solution for the minimization version of (3), which is a contradiction. The argument for the maximization version is analogous. \square

Fig. 1 provides a graphical intuition for the proof: We can find the two solutions to $\ell_i(\theta_i) = \ell^* + \frac{1}{2}\chi_\alpha^{2,-1}$ by starting at the minimum θ_i^* of ℓ and then decreasing/increasing θ_i

as much as possible while maintaining a loss that is at most $\ell^* + \frac{1}{2}\chi_\alpha^{2,-1}$. To do so, we automatically need to adjust all other parameters to allow for as much slack as possible to increase θ_i .

Alternating optimization solver. The key insight for our solver is that problem (3) becomes much more efficient if we *only* optimize over θ_i and not over any remaining parameters. In particular, we can re-write the NLL (1) as:

$$\begin{aligned} \tilde{\ell}_i(\theta_i) &= \sum_{j=1}^n -y_{i,j} \cdot \log[p_{i,j}] - (1 - y_{i,j}) \cdot \log[1 - p_{i,j}] \\ &\quad + \frac{1}{2C} \cdot \theta_i^2 + \ell_{-i}, \end{aligned} \quad (4)$$

where ℓ_{-i} is a constant term that does not depend on θ_i . Computing $\tilde{\ell}_i$ only requires $\mathcal{O}(n)$ computations, whereas the full NLL (1) requires $\mathcal{O}(m \cdot n)$. Overall, we obtain the simplified problem:

$$\min_{\theta_i} \pm\theta_i \quad \text{s.t.} \quad \tilde{\ell}_i(\theta_i) \leq \ell^* + \frac{1}{2}\chi_\alpha^{2,-1}. \quad (5)$$

By extension of Theorem 1, solving (5) is equivalent to solving the equation $\tilde{\ell}_i(\theta_i) = \ell^* + \frac{1}{2}\chi_\alpha^{2,-1}$, which we can do efficiently via standard nonlinear equation solvers.

Importantly, our solution of (5) will be sub-optimal according to the original problem (3) because $\tilde{\ell}_i$ only approximates ℓ_i . Therefore, we update $\tilde{\ell}_i$ by keeping θ_i fixed and minimizing over all other parameters $\vec{\theta}_{-i}$ and \vec{b} . Then, we can solve (5) again, and so forth, until convergence. This is our proposed alternating optimization (AO) solver.

Figure 2 shows an illustration of the algorithm. To infer the upper bound θ_i^+ , we start at $\theta_{0,i}^+ = \theta_i^*$, i.e. the optimal value according to the NLL. We obtain the next estimate $\theta_{1,i}^+$ by solving the $-\theta_i$ version of (5) with respect to the current surrogate NLL $\tilde{\ell}_{0,i}^+$. Then, we update all other parameters $\vec{\theta}_{-i}$ and \vec{b} by minimizing the NLL (1), yielding a new surrogate $\tilde{\ell}_{1,i}^+$. Solving (5) again yields the next estimate $\theta_{2,i}^+$. In this example, $\theta_{2,i}^+$ would already be indistinguishable from the optimal θ_i^+ according to (3) because $\tilde{\ell}_{1,i}^+$ and ℓ_i strongly overlap. In general, we can prove that $\theta_{t,i}^+$ converges to θ_i^+ .

THEOREM 2. *Let $\theta_{0,i}^\pm = \theta_i^*$, where $(\vec{\theta}^*, \vec{b}^*)$ minimizes the NLL $\ell(\vec{\theta}, \vec{b})$. Let $\tilde{\ell}_{t,i}^\pm$ be the surrogate NLL (4) for parameters $\min_{\vec{\theta}_{-i}, \vec{b}} \ell(\vec{\theta}, \vec{b})$ for fixed $\theta_i = \theta_{t,i}^\pm$, and let $\theta_{t+1,i}^\pm$ be the solutions of the \mp versions of (5) for $\tilde{\ell}_i = \tilde{\ell}_{t,i}^\pm$. Then, for $C > 0$ and $t \rightarrow \infty$, $\theta_{t,i}^\pm$ converge to solutions of the minimization/maximization version of (3).*

PROOF. *As a notational shorthand, let $L_\alpha := \ell^* + \frac{1}{2}\chi_\alpha^{2,-1}$. For simplicity, we only consider $\theta_{t,i}^+$ here; the proof for $\theta_{t,i}^-$ is analogous. First, observe that, for all t , we have $\tilde{\ell}_{t,i}^+(\theta_{t,i}^+) = \ell_i(\theta_{t,i}^+)$ by construction, and we have $\tilde{\ell}_{t,i}^+(\theta_{t+1,i}^+) = L_\alpha$, otherwise $\theta_{t+1,i}^+$ would not be a solution to (5). Further, by definition, we have $\tilde{\ell}_{t,i}^+(\theta_i) \geq \ell_i(\theta_i)$ for all t and all $\theta_i \in \mathbb{R}$. Therefore, we obtain $\tilde{\ell}_{t,i}^+(\theta_{t,i}^+) = \ell_i(\theta_{t,i}^+) \leq \tilde{\ell}_{t-1,i}^+(\theta_{t,i}^+) = L_\alpha$.*

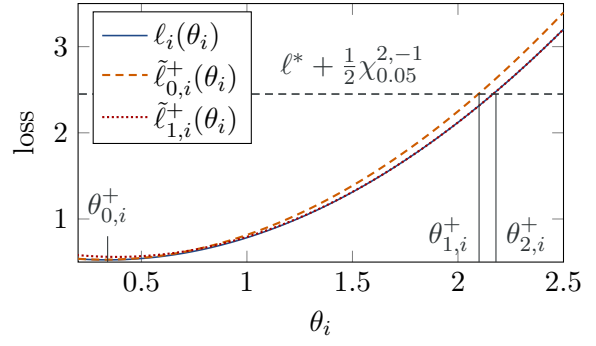


Figure 2: Illustration of the alternating optimization algorithm. We start with the optimum value for $\theta_{0,i}^+ = \theta_i^*$, then maximize θ_i via (5), yielding $\theta_{1,i}^+$, then update $\tilde{\ell}_i$, then maximize θ_i via (5) again, and so forth.

Accordingly, whenever we solve (5) in step $t + 1$, $\theta_{t,i}^+$ is a feasible initial point. Therefore, $\theta_{t+1,i}^+ \geq \theta_{t,i}^+$.

Whenever $\theta_{t+1,i}^+ = \theta_{t,i}^+$, we have $\tilde{\ell}_{t+1,i}^+ = \tilde{\ell}_{t,i}^+$ and, thus, $\theta_{t,i}^+ = \theta_{t+1,i}^+ = \theta_{t+2,i}^+ = \dots$, that is, we have a fixed point. Further, we have $\ell_i(\theta_{t+1,i}^+) = \tilde{\ell}_{t+1,i}^+(\theta_{t+1,i}^+) = \tilde{\ell}_{t+1,i}^+(\theta_{t,i}^+) = L_\alpha$, which is a solution to the maximization version of (3). Before the fixed point, the sequence $\theta_{0,i}^+, \theta_{1,i}^+, \dots$ is strictly increasing. Since ℓ_i is convex with a minimum at $\theta_{0,i}^+$, the sequence $\ell_i(\theta_{0,i}^+), \ell_i(\theta_{1,i}^+), \dots$ is also increasing. For $C > 0$, it would eventually grow beyond bounds due to the regularization term, but since it is upper-bounded by L_α , it must converge to L_α and, hence, $\theta_{t,i}^+$ must converge to θ_i^+ . \square

While Theorem 2 only holds in the limit, very few steps t suffice for a close-to-optimal solution in practice (e.g., Figure 2). We investigate this issue in more detail in our experiments.

3. EXPERIMENTS

In our experiments, we simulate synthetic data from a ground-truth IRT model with ability and difficulties sampled from a standard normal distribution. We varied the number of students m in the range $\{30, 50, 100, 500\}$ and the number of items n in the range $\{10, 20\}$ (similar to the protocol of [3]). For each model, we repeated the sampling 10 times to assess variation. In each repeat, we computed confidence intervals for θ_i at $\alpha = 0.05$ via Wald’s method, the likelihood profile-method via solving $\ell_i(\theta_i) = \ell^* + \frac{1}{2}\chi_\alpha^{2,-1}$ for each i (LP), a log-barrier solver (barrier) for (3), and our proposed alternating optimization scheme (Theorem 2) for $t = 1$ (AO(1)), $t = 2$ (AO(2)), and $t = 3$ (AO(3)) steps. All experimental code is available at https://github.com/bpaassen/ability_bounds.

Coverage. Table 1 shows the coverage rates of all methods, that is, the rate at which the ground truth θ_i value was included in the confidence interval $[\theta_i^-, \theta_i^+]$ [3]. For $\alpha = 0.05$, the coverage rate should be as close as possible to 0.95. We observe that Wald’s method selects too large confidence intervals, yielding rates close to 100%. The barrier method

Table 1: Coverage rates of all methods across experimental conditions

m	n	Wald	LP	barrier	AO(1)	AO(2)	AO(3)
30	10	0.983 ± 0.022	0.943 ± 0.054	0.910 ± 0.056	0.943 ± 0.054	0.943 ± 0.054	0.943 ± 0.054
30	20	1.000 ± 0.000	0.943 ± 0.021	0.083 ± 0.050	0.940 ± 0.025	0.943 ± 0.021	0.943 ± 0.021
50	10	0.998 ± 0.006	0.926 ± 0.041	0.884 ± 0.047	0.920 ± 0.041	0.926 ± 0.041	0.926 ± 0.041
50	20	0.998 ± 0.006	0.942 ± 0.039	0.098 ± 0.039	0.938 ± 0.040	0.942 ± 0.039	0.942 ± 0.039
100	10	0.998 ± 0.004	0.934 ± 0.028	0.892 ± 0.029	0.933 ± 0.027	0.934 ± 0.028	0.934 ± 0.028
100	20	0.999 ± 0.003	0.937 ± 0.030	0.059 ± 0.012	0.933 ± 0.031	0.937 ± 0.030	0.937 ± 0.030
500	10	0.998 ± 0.002	0.940 ± 0.011	0.898 ± 0.015	0.939 ± 0.010	0.940 ± 0.011	0.940 ± 0.011
500	20	1.000 ± 0.001	0.949 ± 0.008	0.097 ± 0.016	0.948 ± 0.008	0.949 ± 0.008	0.949 ± 0.008

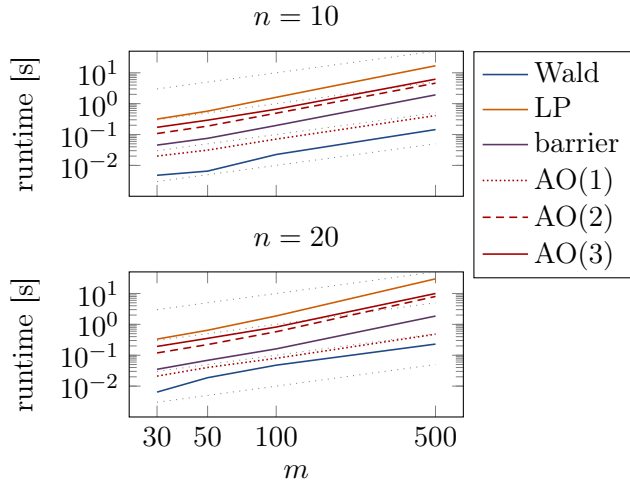


Figure 3: Log-Log plots for the runtime in seconds versus m for $n = 10$ (top) and $n = 20$ (bottom). Dotted, gray lines show linear functions at .1, 1, 10, and 100ms per student.

selects smaller confidence intervals, yielding rates around 90% for $n = 10$ items. If we choose $n = 20$ items, the barrier method becomes numerically unstable and the coverage rates degrade (smaller 10%). The likelihood profile (LP) method consistently achieves rates between 92% and 95% and thus is closest to the desired value of 95%. Alternating optimization achieves indistinguishable results to LP at 3 significant digits for $t \geq 2$ (AO(2) and AO(3)).

Runtime. Figure 3 displays the time it took to compute confidence intervals for all students in every experimental settings in log-log plots. Dotted, gray lines are linear reference functions at .1, 1, 10, and 100ms per student. Using linear fits, we find that Wald’s method is fastest (at about .3ms per student), followed by AO(1) (about .8ms per student), barrier (about 4ms per student), AO(2) (about 9ms per student), AO(3) (about 12ms per student), and finally LP (about 37ms per student). Accordingly, AO(1) is roughly 50x faster than LP, and AO(2) is roughly 4x faster.

4. DISCUSSION AND CONCLUSION

In this paper, we introduced a new solver for confidence intervals on item response theory parameters via the likelihood profile method. In particular, we found that our

alternating optimization solver was 4-50 times faster than the existing solver while achieving almost indistinguishable results. For anyone who seeks to compute confidence intervals, this should provide massive speedups in practice. More generally, though, we hope that our new solver can help the community to respond to an emerging need in educational data mining: more and more, policy makers and the general public expect machine learning models to provide explanations for their decisions. This is exemplified by recent policy initiatives in the European Union for a right to explanation and a risk-based approach to regulate artificial intelligence. Grading student ability—like in item response theory—will likely be under increasing scrutiny in the years to come. As such, we believe that it is crucial to quantify a model’s uncertainty precisely, which our solver can help to do.

We also provide a key theoretical insight in our paper: Traditionally, confidence intervals express the range of parameter values which is likely to contain the ‘true’ value. We showed that the likelihood profile method for confidence intervals can also be interpreted as an optimization problem: We try to find the minimum/maximum ability value for a student which is still consistent with a high likelihood of the data. This interpretation provides a new way to explain an ability estimate: The upper bound of our confidence interval is the highest possible grade we can give to a student while still being consistent with the responses they provided.

Beyond this paper, there remains ample room for future work. Our evaluation has only covered synthetic data to validate the runtime advantage and the closeness to existing methods. Future work could investigate how large confidence intervals tend to be in practical scenarios. Further, our experiments indicated that the size of our confidence is still larger than it would need to be to cover the ‘true’ ability value. Future work could try to find new methods to compute confidence intervals which are more precise. In particular, it may be promising to investigate combinations with recently proposed models for variational item response theory [14].

Acknowledgements

Funding by the German Ministry for Education and Research (BMBF) under grant number 21INVI1403 (project KIPerWeb) is gratefully acknowledged.

5. REFERENCES

[1] F. Baker and S.-H. Kim. *Item Response Theory: Parameter Estimation Techniques*. CRC Press, Boca

- Raton, FL, USA, 2 edition, 2004.
- [2] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, UK, 2009.
 - [3] R. P. Chalmers, J. Pek, and Y. Liu. Profile-likelihood confidence intervals in item response theory models. *Multivariate Behavioral Research*, 52(5):533–550, 2017.
 - [4] A. Doebler, P. Doebler, and H. Holling. Optimal and most exact confidence intervals for person parameters in item response theory models. *Psychometrika*, 78(1):98–115, 2013.
 - [5] S. Embretson and S. Reise. *Item response theory for psychologists*. Psychology Press, New York, NY, USA, 2000.
 - [6] European Commission. Laying down harmonised rules on artificial intelligence and amending certain union legislative acts, 2021.
 - [7] C. Göpfert, L. Pfannschmidt, J. P. Göpfert, and B. Hammer. Interpretation of linear classifiers by means of feature relevance bounds. *Neurocomputing*, 298:69–79, 2018.
 - [8] R. Hambleton and H. Swaminathan. *Item response theory: Principles and applications*. Springer Science+Business Media, New York, NY, USA, 1985.
 - [9] M. E. Kaminski. The right to explanation, explained. *Berkeley Technology Law Journal*, 34:190–218, 2019.
 - [10] R. P. McDonald. A basis for multidimensional item response theory. *Applied Psychological Measurement*, 24(2):99–114, 2000.
 - [11] K. Niemeijer, R. Feskens, G. Kreml, J. Koops, and M. J. S. Brinkhuis. Constructing and predicting school advice for academic achievement: A comparison of item response theory and machine learning techniques. In *Proceedings of the Tenth International Conference on Learning Analytics & Knowledge (LAK)*, page 462–471, New York, NY, USA, 2020. Association for Computing Machinery.
 - [12] P. I. Pavlik, H. Cen, and K. R. Koedinger. Performance factors analysis –a new alternative to knowledge tracing. In *Proceedings of the International Conference on Artificial Intelligence in Education (AIED)*, page 531–538, 2009.
 - [13] K. Pliakos, S.-H. Joo, J. Y. Park, F. Cornillie, C. Vens, and W. Van den Noortgate. Integrating machine learning into item response theory for addressing the cold start problem in adaptive learning systems. *Computers & Education*, 137:91–103, 2019.
 - [14] M. Wu, R. L. Davis, B. W. Domingue, C. Piech, and N. Goodman. Variational item response theory: Fast, accurate, and expressive. In A. Rafferty, J. Whitehill, V. Cavalli-Sforza, and C. Romero, editors, *Proceedings of the 13th International Conference on Educational Data Mining (EDM)*, pages 257–268, 2020.

Towards the understanding of cultural differences in between gamification preferences: A data-driven comparison between the US and Brazil

Armando Toda
Computer Science Dept.
Durham University
Institute of Mathematics and
Computer Science - University
of Sao Paulo
armando.toda@usp.br

Ana C. T. Klock
Gamification Group
Tampere University
Tampere - Finland
actklock@gmail.com

Filipe D. Pereira
Computer Science Dept.
Federal University of Roraima
Roraima - Brazil
filipedwan@icomp.ufam.edu.br

Luiz A. Rodrigues
Institute of Mathematics and
Computer Science
University of Sao Paulo
Sao Carlos - Brazil
lalrodrigues@usp.br

Paula T. Palomino
Institute of Mathematics and
Computer Science
University of Sao Paulo
Sao Carlos - Brazil
paulatpalomino@usp.br

Vinicius Lopes
Institute of Mathematics and
Computer Science
University of Sao Paulo
Sao Carlos - Brazil
lopsvinciuss@gmail.com

ABSTRACT

Research on tailored gamification has shifted from analysing single students' characteristics (e.g. gender or behavioural profiles) to multiple characteristics and how they are influenced by gamification (e.g. context and system log usage). Yet, few studies have been conducted that are concerned with culture, which influences many of the students' characteristics and, consequently, their learning performance. In order to provide a better gamified experience, it is important to understand culture and how it impacts on students' perceived importance of gamification elements, so these elements can be adapted to specific cultures. To this end, we conducted an exploratory study using Association Rule Mining, to explore how the cultural landscape (country) influences the students' behaviour and perceived importance of gamification elements in educational environments. We collected and analysed data ($N = 1296$) from two different countries, Brazil ($N = 428$) and the United States ($N = 868$) and identified significant differences between the perceived importance of some gamification elements.

Keywords

gamification, association rules, culture, personalisation

1. INTRODUCTION

Personalised gamification in learning environments has become a trend in the past few years [7, 6]. This field of study focuses on enhancing existing gamification approaches with

A. Toda, A. Klock, F. D. Pereira, L. A. Rodrigues, P. T. Palomino, V. Lopes, C. Stewart, E. H. T. Oliveira, I. Gasparini, S. Isotani, and A. Cristea. Towards the understanding of cultural differences in between gamification preferences: A data-driven comparison between the US and Brazil. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 560–564, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853030>

user-centred and personalised design, tailored to the specific characteristics of the user, which will have an impact on their perceived satisfaction, engagement, and motivation, when using a gamified system [4]. Tailored gamification engaged the interest of researchers and educational professionals, due to facilitating positive effects associated with it, such as motivation and performance increase [12].

Recent studies focused on the use of personalised approaches using student demographics (e.g. gender) [21], gamer profiles (e.g. using the HEXAD [22] model), and gaming preferences (e.g. users' favourite game genres or elements) [14] as part of the gamification design.

Whilst these recent studies demonstrated that personalised gamification tends to have a positive influence on students' motivation and performance, there is still no consensus on what kind of personalisation and which attributes should be considered, since most studies focus only on single factors [16]. Literature on personalised gamification points to a lack of studies that deal with other learners' characteristics that can also influence their perception and interaction with gamification; one such important characteristic being *culture* [6, 13].

Culture is not a trivial concept to define; according to [18], culture is an evolving cognitive structure, which influences the behaviour of members of a given group. Features included in the schema of this structure are those that influence and are influenced by the geographical location of an individual and educational contexts they are exposed to [17]. It is this aspect of 'culture' that we will seek to investigate within the rest of this paper. Recently, culture's importance to gamification has come to the fore [2, 24, 19]. In spite of this surge in interest, according to [19], most existing gamification studies that address education and culture are focused on language learning, rather than on understanding how culture can impact on gamification design.

Based on what has been exposed, this paper aims to answer the following research question "Does culture impact on the importance of gamification elements?". We answer this research question, by conducting a relatively large and representative¹ quantitative study, analysing the perceived importance of gamification elements in two different countries (United States and Brazil). Our results provide empirical evidence on the differences between the perceived importance of gamification elements for these two different countries, which can guide designers when considering national culture as a set of variables² in gamification design for learning environments, and also guide researchers to conduct new research in this field. In summary, our main contributions are new insights on which game elements for education are preferred in different countries.

2. METHODS

Considering our main research question, we designed an initial sub-research question to identify if a certain gamification element is more important in a country than in another (here, Brazil versus US). We use <gamification element> for any element from the Taxonomy of Gamification Elements for Education (TGEE) proposed by [20], while the 'country' refers to data collected from people living in a given country.

We opted for the TGEE [20] due to it being the first classification created and evaluated for educational purposes. TGEE is composed of 21 gamification elements, alongside its synonyms, descriptions, and examples of use. These elements are classified in 5 categories that deal with students' performance, sociability, personal information, and experience, and the environment ecosystem.

Through analysing the perceived importance of gamification elements in different countries, we can infer how the culture can influence the design of personalised gamified applications for education. To conduct this research, first we used inferential statistics, to test our assumption that the elements' perceived importance differs, depending on the country of residence. To do so, we used the Mann-Whitney U test (due to the non-parametric and independent nature of our data [11]) and Cliff's delta, to understand to what extent perceptions differ (i.e., the effect size) [10]. Following this we conduct an exploratory analysis, using unsupervised learning algorithms and descriptive/inferential statistics to provide inferences that can be explored deeply in future studies. The data-driven pipeline consists of data collection, filtering, and analysis, further explained in the respective sections.

To collect our data, we relied on the survey method, due to its low cost and other benefits (e.g. speed of obtaining answers) [8]. The survey used in this study was designed in Google forms and consists of two parts. The first collects self-reported demographic information about the respondents, such as gender, country (of residence) and some information regarding their 'gamer' status, such as: years

¹According to Sample Size calculator, our sample is representative for both US and Brazil. Link: <https://www.checkmarket.com/sample-size-calculator/>

²It is important to note that culture itself cannot be defined in one single variable (e.g. country), but a set of characteristics from that might be associated with the place of origin, social group, etc..

playing games, hours per week spent playing, favourite game genres and favourite setting (single or multiplayer games). Whilst it is known that games and gamification are not the same, however, recent studies [15] demonstrated that gaming characteristics of users do influence their gamification acceptance. Previous literature also showed that students' player profile might additionally influence their performance when using gamified learning environments. Based on these premises, we collect the participants' gaming culture information, to establish how it influences the perceived importance of the gamification elements. The second part of the survey consists of 21 gamification elements that were proposed for educational contexts, where we asked what the respondents' perceived importance for each of the 21 gamification elements was, on a Likert scale [9] from 1 to 5, where 1 meant "not important at all" and 5 "extremely important".

The recruitment of respondents occurred via (1) Amazon Mechanical Turk, demonstrated to be an effective platform to obtain a representative sample of answers especially in the US [3], and (2) social networks, which allow us to reach a broad audience and can also be useful tools for survey recruitment [3], social networks were used mainly to collect data from Brazil. We left the survey open for answers for 3 months. The first page of the survey contained the informed consent form³, which the participant needed to agree to, in order to participate in the study, as well as the information regarding the study's objectives, as suggested in [5].

After preparing our dataset, we used descriptive statistics and ARM to explore it. Descriptive statistics allow us to understand the significant differences between countries, which in addition can support us in answering. ARM consists of verifying the associations between the data, by presenting *if-then-else* clauses that are explained using a given set of metrics. It is mainly used in market-basket analysis, to identify possible combinations of elements [1]. In this work, we will consider support (related to the frequency of a given item), confidence (related to the strength of the clause based on its frequency), and lift (which measures the independence of the items in the clause) as the main metrics for ARM.

3. RESULTS AND DISCUSSIONS

In this section, we present the results of our analyses using the given dataset. Initially, our raw data consisted of 1952 answers; after applying our filters, as described in the previous section, we obtained a total of 1296 respondents who live in Brazil ($n = 428$; 33%) or the US ($n = 868$; 67%). Considering the country distribution and genders (Table 1), 428 are from Brazil (33%) and 868 from the US (67%). Among the respondents, 447 identify themselves as female (34.5%), 839 as male (64.7%), nine as non-binary (0.7%) and one person preferred not to disclose (0.1%).

Considering their age and time spent on games distribution: the minimum age (in years) is 16, with maximum age

³The consent allows us to analyse and publish the answers of the respondents. The data collected within this study is in accordance with the General Data Protection Regulation (GDPR⁴) and any kind of personal information was removed prior to the analysis. This research was also approved by an ethical committee 42598620.0.0000.5464 at University of Sao Paulo.

Table 1: Gender distribution

Gender distribution	Brazil		US		Total
	N	%	N	%	
Female	81	18.93	366	42.17	447
Male	344	80.37	495	57.03	839
Non-Binary	3	0.70	6	0.69	9
Prefer not to disclose	0	0	1	0.12	1
Total	428	100	868	100	1296

77. The medium age of the population is 29.7 with a standard deviation of 10. Considering the experience and gaming preferences of our population, they had an average of 18.7 years of playing games ($SD = 8.2$), and an average of 14 hours per week ($SD = 13.8$). The top 3 favourite game genres were RPG (499 answers, 38.5%), adventure (259 answers, 20%) and strategy (186 answers, 14.4%).

The favourite setting (Table 2) of the respondents was single player games (898 answers, 69.3%), followed by multiplayer games (398, 30.7%). In Brazil, the top 3 favourite game genres were RPG ($N = 182$, 42.5%), followed by Adventure ($N = 66$, 15.4%), and Strategy ($N = 65$, 15.2%). As for the US, they considered RPG ($N = 317$, 36.5%), followed again by Adventure ($N = 193$, 22.24%), and Strategy ($N = 121$, 13.94%).

Table 2: Favourite setting distribution

Favourite setting	Brazil		US		Total
	N	%	N	%	
Multiplayer	137	32,009	261	30,069	398
Singleplayer	291	67,991	607	69,931	898
Total	428	100	868	100	1296

When applying our sub-research question formula to each of the elements, we can see in Table 3 that only 6 elements showed a significant difference ($p < 0.05$, after corrected with False discovery rate (FDR)). We also applied Cliff’s Delta to measure the effect size (since this delta can be used to measure the effect size of different size populations) and obtained small effect size results on the same 6 elements [23].

Considering the results seen in Table 3, we can observe that 6 elements (Cooperation, Novelty, Stats, Sensation, Narrative, and Storytelling) had a significant difference in the hypothesis tests ($p < 0.05$), and a small effect-size (up to 0.3). In practice, this means that each of the analysed countries consider these elements to a different degree of importance. The low effect-size indicates that these element differences may pose a significant impact when tied with other elements considering the respondents’ country; this must be confirmed with empirical studies using these elements.

As for the Association Rules, when mining all the rules that contain the variable country, we found 937 rules. In this work, we considered only rules that contained a gamification element and that satisfies the following conditions: (A) *confidence* > 0.8 ; and (B) *lift* > 1.1 , similar as seen in [21].

Considering our conditions, we found 44 rules for condition (A); and 295 for condition (B). When considering the in-

tersection between condition A and B, we found 40 rules ($A \cap B = 40$). The strongest rule (rule 3275, confidence = 0,93; lift = 1,99) that was mined was associating Brazil with Objective, Storytelling, and Narrative. The following rules also associated Brazil with different gamification elements as Progression (rule 3279), Sensation (rule 3270) and characteristics as favourite setting as single player (rule 3278). The same set of gamification elements (Narrative, Storytelling, Progression, and Objectives) were also associated with the United States. The major difference was in the Sensation element that was found in the strongest rules associated with Brazil (Table 4).

4. CONCLUSIONS AND FUTURE WORKS

This paper presented a data-driven work focusing at exploring cultural differences between Brazil and the US regarding their perceived preferences on gamification elements. We found significant and interesting associations and groups of gamification elements that can be used in educational environments (both virtual and non-virtual). These can also be used as input for adaptive gamification environments as a new set of variables that can be grouped by the country of origin of the student.

In summary, we provide as the main contributions of our work the first empirical evidence on the impact of culture in the perceived importance of gamification elements, comparing Brazil and the US, as well as providing new strategies based on respondents’ country of origin and others easy-to-obtain characteristics that can be used in adaptive gamified learning environments.

In future works, we intend to expand the research by applying other unsupervised algorithms, and investigating other sub-cultures within these countries and identify different and similar patterns. We also intend to expand the concept of culture and not associate it only with a country, but also including the region the respondent was raised and live nowadays, as well as other variables that can be related to their education level, or gaming preferences that might have an influence on their perceived importance of gamification elements. Another future work is related to the implementation of these strategies in a real educational environment. Finally, it would be interesting to also analyse other countries and see the differences and similarities between patterns.

5. ACKNOWLEDGMENTS

This research was partially funded by the following organizations: Brazilian National Council for Scientific and Technological Development (CNPq)—processes 141859/2019-9, 163932/2020-4, 308458/2020-6, and 308513/2020-7; Coordination for the Improvement of Higher Education Personnel (CAPES)—Finance Code 001; and São Paulo State Research Support Foundation (FAPESP)—processes 2018/15917-0 and 2013/07375-0. Additionally, this research was carried out within the scope of the Samsung-UFAM Project for Education and Research (SUPER), according to Article 48 of Decree no 6.008/2006 (SUFRAMA), and partially funded by Samsung Electronics of Amazonia Ltda., under the terms of Federal Law no 8.387/1991, through agreements 001/2020 and 003/2019, signed with Federal University of Amazonas and FAEPI, Brazil.

Table 3: Summary of gamification elements. Elements were organised based on the order they appeared in the survey.

Element	Brazil		US		Mann-whitney P-value	FDR	Cliff's Delta Effect Size
	Mean	SD	Mean	SD			
point	3.4	1.23	3.34	1.23	0.46	0.48	0.03
level	4.02	0.96	3.87	1.06	0.03	0.08	0.07
cooperation	3.55	1.16	3.31	1.25	0	0.01	0.10
competition	3.05	1.29	3.17	1.34	0.14	0.21	-0.05
renovation	3.64	1.01	3.52	1.05	0.09	0.16	0.06
progression	4.41	0.82	4.35	0.81	0.08	0.16	0.05
objectives	4.35	0.83	4.25	0.88	0.05	0.11	0.06
puzzle	3.82	0.99	3.73	1.12	0.35	0.41	0.031
novelty	3.98	0.92	3.72	1.01	0	0	0.14
chance	3.19	1.20	3.08	1.12	0.10	0.17	0.05
social_pressure	2.59	1.22	2.50	1.25	0.16	0.21	0.05
acknowledgement	3.55	1.18	3.45	1.21	0.15	0.21	0.05
stats	4.05	0.97	3.82	1.07	0	0	0.12
rarity	3.38	1.18	3.30	1.19	0.39	0.43	0.03
imposed_choice	3.14	1.09	3.24	1.1	0.09	0.16	-0.06
time_pressure	2.58	1.15	2.64	1.24	0.51	0.51	-0.02
economy	3.14	1.28	3.2	1.26	0.36	0.41	-0.03
sensation	4.31	0.9	3.80	1.1	0	0	0.27
reputation	3.02	1.18	3.16	1.2	0.04	0.11	-0.07
narrative	4.41	0.91	3.93	1.12	0	0	0.26
storytelling	4.35	0.91	4.05	1.13	0	0	0.15

Table 4: Top 15 rules found in ARM.

Rule ID	Left-hand side	Right-hand side	Support	Confidence	Coverage	Lift
3275	{country=Brazil,objectives=5,storytelling=5}	{narrative=5}	0.11	0.93	0.11	1.99
3279	{country=Brazil,progression=5,storytelling=5}	{narrative=5}	0.11	0.88	0.12	1.88
3269	{country=Brazil,sensation=5,narrative=5}	{storytelling=5}	0.10	0.87	0.12	1.74
3287	{country=Brazil,fav_setting=Singleplayer,storytelling=5}	{narrative=5}	0.12	0.87	0.14	1.86
3270	{country=Brazil,sensation=5,storytelling=5}	{narrative=5}	0.10	0.87	0.12	1.85
3273	{country=Brazil,objectives=5,narrative=5}	{storytelling=5}	0.11	0.87	0.12	1.73
2321	{country=Brazil,storytelling=5}	{narrative=5}	0.17	0.86	0.19	1.84
3374	{gender=Female,fav_setting=Singleplayer,progression=5}	{country=United States}	0.10	0.85	0.12	1.27
3310	{country=Brazil,fav_setting=Singleplayer,progression=5}	{gender=Male}	0.10	0.85	0.12	1.32
2692	{gender=Female,renovation=4}	{country=United States}	0.11	0.85	0.13	1.26
3283	{gender=Male,country=Brazil,storytelling=5}	{narrative=5}	0.13	0.85	0.15	1.81
2732	{gender=Female,fav_setting=Singleplayer}	{country=United States}	0.21	0.84	0.24	1.26
3765	{country=United States,fav_setting=Singleplayer,progression=5,narrative=5}	{storytelling=5}	0.11	0.84	0.13	1.68
2695	{gender=Female,novelty=4}	{country=United States}	0.11	0.84	0.13	1.25
1573	{country=United States,fav_genre=Adventure}	{fav_setting=Singleplayer}	0.12	0.82	0.15	1.19

6. ADDITIONAL AUTHORS

Additional authors: **Craig D. Stewart** from Durham University, **Elaine Harada Teixeira de Oliveira** from Federal University of Amazonas, **Isabela Gasparini** from Santa Catarina State University, **Seiji Isotani** from University of Sao Paulo, and **Alexandra I. Cristea** from Durham University.

7. REFERENCES

- [1] R. Agrawal, T. Imieliński, and A. Swami. Mining association rules between sets of items in large databases, 1993.
- [2] A. AlMarshedi, V. Wanick, and G. B. Wills. Gamification and Behaviour. In *Gamification*, pages 3–18. 2016.
- [3] F. Bentley, K. O. Neill, K. Quehl, and D. Lottridge. Exploring the Quality, Efficiency, and Representative Nature of Responses Across Multiple

- Survey Panels. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–12, New York, NY, USA, apr 2020. ACM.
- [4] M. Böckle, J. Novak, and M. Bick. Towards Adaptive Gamification: A Synthesis of Current Developments. In *Twenty-Fifth European Conference on Information Systems (ECIS)*, jun 2017.
- [5] L. Gelinas, R. Pierce, S. Winkler, I. G. Cohen, H. F. Lynch, and B. E. Bierer. Using Social Media as a Research Recruitment Tool: Ethical Issues and Recommendations. <https://doi.org/10.1080/15265161.2016.1276644>, 17(3):3–14, mar 2017.
- [6] A. C. T. Klock, I. Gasparini, M. S. Pimenta, and J. Hamari. Tailored gamification: A review of literature. *International Journal of Human-Computer Studies*, page 102495, jun 2020.
- [7] E. Lavoue, B. Monterrat, M. Desmarais, S. S. George, É. Lavoué, B. Monterrat, M. Desmarais, and S. S. George. Adaptive Gamification for Learning Environments. *IEEE Transactions on Learning Technologies*, 12(1):16–28, jan 2018.
- [8] J. Lazar, J. H. Feng, and H. Hochheiser. *Research methods in human-computer interaction*. Morgan Kaufmann, 2nd edition, 2017.
- [9] R. Likert. A technique for the measurement of attitudes. *Archives of Psychology*, 140(140):44–53, 1932.
- [10] G. Macbeth, E. Razumiejczyk, and R. D. Ledesma. Cliff’s delta calculator: A non-parametric effect size program for two groups of observations. *Universitas Psychologica*, 10(2):545–555, 2011.
- [11] H. B. Mann and D. R. Whitney. On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other. *The Annals of Mathematical Statistics*, 18(1):50–60, jun 1947.
- [12] W. Oliveira and I. I. Bittencourt. *Tailored Gamification to Educational Technologies*. Springer Singapore, Singapore, 2019.
- [13] P. T. Palomino, A. M. Toda, L. Rodrigues, W. Oliveira, and S. Isotani. From the Lack of Engagement to Motivation: Gamification Strategies to Enhance Users Learning Experiences. *SBC – Proceedings of SBGames 2020*, pages 1127–1130, 2020.
- [14] L. Rodrigues, P. T. Palomino, A. M. Toda, A. C. T. Klock, W. Oliveira, A. P. Avila-Santos, I. Gasparini, and S. Isotani. Personalization Improves Gamification: Evidence from a Mixed-Methods Study. *Proc. ACM Hum.-Comput. Interact.*, 5(CHI PLAY), oct 2021.
- [15] L. Rodrigues, F. Pereira, A. Toda, P. Palomino, W. Oliveira, M. Pessoa, L. Carvalho, D. Oliveira, E. Oliveira, A. Cristea, and S. Isotani. Are they learning or playing? moderator conditions of gamification’s success in programming classrooms. *ACM Trans. Comput. Educ.*, nov 2021. Just Accepted.
- [16] L. Rodrigues, A. M. Toda, P. T. Palomino, W. Oliveira, and S. Isotani. Personalized gamification: A literature review of outcomes, experiments, and approaches. *ACM International Conference Proceeding Series*, pages 699–706, oct 2020.
- [17] I. Savard, J. Bourdeau, and G. Paquette. Considering cultural variables in the instructional design process: A knowledge-based advisor system. *Computers and Education*, 145:103722, feb 2020.
- [18] I. Savard and R. Mizoguchi. Context or culture: what is the difference?, dec 2019.
- [19] A. Toda, A. C. T. Klock, P. T. Palomino, L. Rodrigues, W. Oliveira, C. Stewart, A. I. Cristea, I. Gasparini, and S. Isotani. GamiCSM: Relating education, culture and gamification - a link between worlds. In *XIX Brazilian Symposium on Human Factors in Computing Systems*, page In press., Diamantina, 2020.
- [20] A. M. Toda, W. Oliveira, A. C. Klock, P. T. Palomino, M. Pimenta, I. Gasparini, L. Shi, I. Bittencourt, S. Isotani, A. I. Cristea, L. Shi, I. Gasparini, S. Isotani, A. I. Cristea, L. Shi, I. Bittencourt, S. Isotani, and A. I. Cristea. A Taxonomy of Game Elements for Gamification in Educational Contexts: Proposal and Evaluation. In *IEEE 19th International Conference on Advanced Learning Technologies (ICALT)*, pages 84–88, 2019.
- [21] A. M. Toda, W. Oliveira, L. Shi, I. Bittencourt, S. Isotani, and A. Cristea. Planning Gamification Strategies based on User Characteristics and DM : A Gender-based Case Study. In *Proceedings of the Educational Data Mining 2019 conference*, number i, pages 438 – 443, Montréal, may 2019.
- [22] G. F. Tondello, R. R. Wehbe, L. Diamond, M. Busch, A. Marczewski, and L. E. Nacke. The Gamification User Types Hexad Scale. In *Proceedings of the 2016 Annual Symposium on Computer-Human Interaction in Play - CHI PLAY ’16*, pages 229–243, New York, New York, USA, 2016. ACM Press.
- [23] A. Vargha and H. D. Delaney. A Critique and Improvement of the CL Common Language Effect Size Statistics of McGraw and Wong. <http://dx.doi.org/10.3102/10769986025002101>, 25(2):101–132, nov 2016.
- [24] R. Wellington. Context to Culture for Gamification HCI Requirements: Familiarity and Enculturation. In *Gamification in Education and Business*, pages 151–163. Springer International Publishing, Cham, 2015.

Supervised Machine Learning for Modelling STEM Career and Education Interest in Irish School Children

Annika Lindh
Maynooth University
annika.lindh@mu.ie

Keith Quille
Technological University Dublin
keith.quille@tudublin.ie

Aidan Mooney
Maynooth University
aidan.mooney@mu.ie

Kevin Marshall
Microsoft
kevmar@microsoft.com

Katriona O’Sullivan
Maynooth University
katriona.osullivan@mu.ie

ABSTRACT

The number of unfilled jobs in Science, Technology, Engineering and Mathematics (STEM) is predicted to rise while young people’s interest in STEM careers and education is declining. Efforts to understand this decline have identified some potentially contributing factors based on statistical correlation analysis. However, these correlations can sometimes have relatively low effect-sizes. In these cases, Machine Learning (ML) techniques may provide an alternative by uncovering more complex patterns that provide stronger predictive accuracy. In this pilot study of Irish school children aged 9-13, supervised ML techniques were applied to model interest in pursuing education and careers in STEM fields. Despite the rather low coefficients from Pearson Correlation, the ML techniques were able to predict an individual’s interest in STEM careers and education with accuracies of 72.79% and 79.88% respectively. Our results suggest that ML techniques could be an important tool in understanding young people’s interest in STEM careers and education by providing models that derive more complex relationships.

Keywords

STEM Attitudes, Machine Learning, Educational Data Mining, STEM Interest in Ireland

1. INTRODUCTION

The importance of Science, Technology, Engineering and Mathematics (STEM) skills is increasing, both in our everyday lives and in the demands on our future workforce [5]. Meanwhile, current educational systems are unable to keep up with this increasing demand; STEM courses tend to suffer from high drop-out rates [17] and only around half of STEM students go on to pursue STEM careers [3]. Coupled with a decrease in young people’s interest in STEM [1, 6, 7], concerns have been raised about the increasing skill-gap; the STEM Education Policy Statement 2017–2026 from the Irish Department of Further and Higher Education, Research, Innovation and Science [5] highlights both the economic and personal consequences in a world that relies on modern technologies.

A. Lindh, K. Quille, A. Mooney, K. Marshall, and K. O’Sullivan. Supervised machine learning for modelling STEM career and education interest in Irish school children. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 565–570, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853026>

An important part of addressing this skill shortage is to understand the underlying factors that drive young people’s interest in STEM fields. Previous studies have employed statistical techniques to identify correlated attributes, including the student’s gender, grades and school experience [4], their self-efficacy scores [11, 20] and affective stereotypical values about STEM major choices [20]; external factors included parents’ education and STEM knowledge [9, 20], as well as their teacher’s knowledge about STEM [20].

While traditional statistical techniques can provide valuable insights, these techniques are not always sufficient to establish strong correlations, e.g. the Pearson Correlation coefficient for individual attributes may be relatively low in relation to the target variable. Supervised Machine Learning (ML) techniques may offer a solution to this, by modelling complex patterns in the data through more advanced mapping functions that predict the outcome variable.

In this pilot study, five traditional ML algorithms – Logistic Regression, k -Nearest Neighbour, Decision Trees, naïve Bayes and Support Vector Machines (SVM) – were applied to predict interest in STEM careers and education among Irish school children aged 9-13. Further, an epilogue experiment was carried out using a Neural Network (Deep Learning) model, to assess whether this may offer additional benefits over the traditional ML techniques.

The code for the experiments is made publicly available online:

<https://github.com/KeithQuille-TUDublin/Supervised-Machine-Learning-for-Modelling-STEM-Career-and-Education-Interest-in-Irish-School-Childre>

2. DATASET

The data used in this study consist of a sub-set of answers collected during 2020 through an online survey directed at Irish primary school children. Participants were recruited from a random selection of primary schools on the Irish government’s list of national schools. Consent was obtained from parents by providing a consent form via email, to comply with the COVID-19 stay-at-home orders at the time; the filled-out forms were collected by teachers and either emailed back as scanned copies or posted as hard copies.

From the survey’s 48 questions, two yes/no questions were selected as target variables: “*I would like to study STEM in the future*” and “*I am interested in a career in STEM*” (referred to as Career in STEM and Study STEM). 36 of the remaining 46 questions were considered as potential independent variables. (These 38 questions, along with their answer types, are listed in Table 3 in the

Appendix.) Out of the 309 participants who filled out the survey, 255 (82.5%) answered all 38 questions of interest. Future surveys will include validation to prevent missing data, and introduce options such as “Don’t know” or “Not applicable” for all relevant questions. For this study, incomplete instances were not included in the analysis. Gender balance and target variable class balances were measured before and after to ensure no bias was introduced with respect to these factors. Proportions remained highly similar, with 40.1% female before and 40.8% after, with interest in Career in STEM changing from 49.5% positive to 49.4%, and Study STEM changing from 69.6% positive to 70.2%.

2.1 Data Preparation

Two datasets were constructed, each including one of the two target variables (Careers in STEM and Study STEM) along with a subset of the 36 potential independent attributes. The two subsets of the independent variables were selected based on statistical correlation between each attribute and each of the target variables. The reasoning behind this, was to reduce the number of noisy attributes. This initial sub-selection can work well even when the correlation magnitude is relatively low as it still helps to filter out those with very low or no correlation. In this study, Pearson Correlation (PC) and Multiple Regression Analysis (MRA) were employed, which are typical techniques for this purpose. Both techniques were applied independently, so that if two different sets of attributes were identified, each set could be used for developing a separate model. Cut-off values for attribute selection were determined by visually examining bar charts of the ordered absolute coefficients for PC and MRA respectively; the selected cut-off values were based on attribute grouping and the elbow method. Future work will include a more comprehensive analysis of the attribute selection step by including additional considerations, such as p -values.

For the Career in STEM dataset, the selected cut-off values for PC and MRA were 0.2 and 10.00 respectively. Both methods produced the same ten attributes (although the order was different):

- Age
- County
- Do you have family in STEM?
- I am good at projects involving Science Technology Engineering and Maths.
- I would like to participate in more after-school programs in Science Technology.
- Confident to problem solve.
- Confident to do science tasks.
- Homework in Science Technology Engineering and Maths is easy.
- Science Technology Engineering and Math is important.
- On average, how long, per day, do you spend using technology at home?

For Study STEM, PC and MRA both produced the same set of 8 attributes with a threshold of 0.21 and 10.73 respectively:

- Age
- Confident to problem solve.
- Confident to do science tasks.
- Confident to use technology in schoolwork.
- I am good at projects involving Science Technology Engineering and Maths.

- Homework in Science Technology Engineering and Maths is easy.
- I would like to participate in more after-school programs in Science Technology.
- Science Technology Engineering and Math is important.

The cut-off point for the Study STEM data was more ambiguous than for the Career in STEM data. To avoid omitting two potentially beneficial borderline attributes, two separate datasets were constructed: Study STEM A which only includes the top 8 attributes, and Study STEM B which includes two additional attributes based on lowering the thresholds for PC and MRA to 0.18 and 7.83 respectively. These attributes were:

- Do you have family in STEM?
- I am good at using technology and completing coding tasks.

Following the attribute selection process, each selected attribute was assessed for outliers by examining the range and spread of values using standard deviation. No significant outliers or unexpected values were identified.

3. MACHINE LEARNING TECHNIQUES

The Machine Learning (ML) investigations of this study were implemented in Python 3.7, using the Scikit-learn¹ v1.0.2 library for the traditional ML algorithms, and TensorFlow² v2.1 for the epilogue Deep Learning experiment. All code was run on a PC with an Intel Core i9 CPU, 32GB RAM and NVIDIA RTX 2070 Super GPU with 8GB RAM. The following sections briefly discuss each algorithm and how they make predictions about a binary class label.

3.1 Logistic Regression

Logistic Regression (LR) is used to predict the binary class label of a data point by estimating the probability of the positive class based on a set of attributes, without having to meet requirements regarding normal distribution or homogeneity of variance [2]. LR can model non-linear relationship between one or more attributes and the class label, according to the following equations:

$$t_x = \sum_{i=1}^N x_i w_i, \quad P(x) = \frac{1}{1 + e^{-t_x}}$$

where w_i are the learned model weights, x_i is a single attribute and N is the number of attributes. $P(x)$ denotes the probability of the positive class, with $P(x) \geq 0.5$ resulting in a positive prediction.

3.2 k-Nearest Neighbour

k -Nearest Neighbour (k -NN) is used to predict the class label for a new data point based on the class label of known data points that have similar attributes to the new data point. The model selects the k closest neighbours (based on a chosen distance metric) and predicts the class label by majority voting [10, 14]. In this study, a value of $k=3$ was used along with the Euclidean distance.

3.3 Naïve Bayes

Naïve Bayes is based on the Bayes Theorem, which is given by $P(Y|X) = P(X|Y) \cdot \frac{P(Y)}{P(X)}$. This assumes the interdependence between all attributes in the term $P(X|Y)$ whose calculation becomes intractable in practice. Hence, a simplified version, known as naïve

¹ <https://scikit-learn.org/>

² <https://www.tensorflow.org/>

Bayes, is often used where $P(X)$ is assumed to be constant and the attributes are assumed to be conditionally independent. In practice, it has been shown to provide strong predictive performance even when this assumption is violated [13, 14]. For a binary classification task, the probability of the positive class is given by $P(y) = P(Y) \cdot \prod_{i=1}^M P(x_i|Y)$ where $P(Y)$ is the prior probability of class y .

3.4 Decision Trees

Decision Trees provide class predictions through a tree-like flowchart where the next branch is selected based on the value of a single attribute. The cut-off points for the attribute values of these branches are decided based on the optimal splitting of the training data into class labels. Once a leaf-node is reached (i.e. no further branching), a class label is assigned based on the majority class of the training data points that were routed to that node [10, 14]. In this study, binary branching was used based on a measure called GINI Gain which aims to reduce the GINI Impurity calculated by $\sum_{i=1}^c p(y_i) \cdot (1 - p(y_i))$ where $p(y_i)$ denotes the probability of each class label.

3.5 Support Vector Machines

Support Vector Machines (SVM) is a class of algorithms that generates a discriminant function to separate the data points belonging to each class label. The SVM used in this study is a linear SVM that uses Sequential Minimal Optimization [16]. This algorithm is grounded in principles of the optimal hyperplane from statistical learning theory [19]. The optimal hyperplane is found by maximising the perpendicular distance between the closest vector to the hyperplane and the hyperplane itself [8]. Given a dataset $(x_1, y_1), \dots, (x_n, y_n) \in X \times \{\pm 1\}$ where each x_i has been sampled from some space X , the optimal hyperplane can be found by solving the dual-form Lagrangian, which is subject to the constraints $\alpha_i \geq \forall_i$ and $\sum_{i=1}^m \alpha_i y_i = 0$:

$$W(\alpha) = \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j=1}^m \alpha_i \alpha_j y_i y_j (x_i \cdot x_j)$$

4. EVALUATION AND METRICS

To promote generalisable model results, 10-fold cross-validation (10FCV) is considered the gold standard validation techniques for these ML algorithms [10]. 10FCV randomly splits the dataset into (as near as possible) equally sized folds; any number of folds can be chosen, with 10 being the typical number. Training is performed on data from all folds except one which is left for evaluation (i.e. the unseen data), and this process is repeated until the algorithm has been trained and evaluated on each fold. The result from each fold's evaluation are averaged to obtain the final result. Thus, all data is considered for evaluation, while at the same time ensuring that test data is never seen during training. This reduces the risk of both under- and overfitting to the data [10].

The metrics used for this study were *sensitivity*, *specificity*, and *accuracy*, each relating to predictions about two mutually exclusive classes, where “yes” is *positive* class, and “no” the *negative* class. True Positives (TP) are correct predictions of the positive class, while True Negatives (TN) are correct predictions of the negative class. Likewise, False Positives (FP) are incorrect predictions of the positive class, while False Negatives (FN) are incorrect predictions of the negative class. Accuracy refers to the proportion of correct predictions (for both the positive and the negative class) in relation to the total number of predictions. Sensitivity refers to the proportion of correctly predicted positive instances TP , in relation to the total number of positive instances $TP + FN$ in the data. High

sensitivity indicates that most of the positive cases are likely found, so if a negative case is predicted, it is highly likely that it is indeed negative. Specificity refers to the proportion of correctly predicted negative instances TN , in relation to the total number of negative instances $TN + FP$ in the data. High specificity means most of the negative cases are likely caught; thus, any positive predictions are highly likely to indeed be positive.

5. RESULTS

5.1 Career in STEM

As shown in Table 1, the highest accuracy was achieved by the SVM model, followed by Logistic Regression (LR), naïve Bayes (nB), Decision Trees (DT) and k-NN. An ANOVA test was carried out, showing statistically significant ($p < .001$) differences between the accuracies with $F(5,255) = 215.8951$. Standard deviation was calculated after applying the binomial distribution formula.

Table 1. Results for the five traditional ML algorithms on predicting an individual's interest in a STEM career

Algorithm	Accuracy	Sensitivity	Specificity
Logistic Regression	71.84%	72%	72%
SVM	72.79%	75%	70%
k-NN	59.15%	58%	60%
Decision Tree	60.02%	60%	63%
Naïve Bayes	71.50%	81%	63%

There is a clear gap in accuracy between the best three models (all above 70%) and the worst two (accuracy around 60%). A possible explanation is the difference in algorithm types; the top three are known for better handling of higher dimensional data with noisy attributes. While steps were taken during the data preparation phase to reduce the number of attributes and lower the risk of noisy data (see sections 2.1), the results indicate that this is an issue that should still be considered during the model design phase.

Sensitivity and specificity followed a similar trend with respect to performance ranking. LR was the best-balanced prediction model with 72% in both sensitivity and specificity. Meanwhile nB was the most imbalanced with 81% sensitivity and 63% specificity, indicating a strong bias towards making positive predictions.

5.2 Study STEM

For the Study STEM target variable, results are reported for both the Study STEM A and Study STEM B datasets (described in section 2.1). Table 2 shows that for STEM A, naïve Bayes (nB) gave the best accuracy, followed by SVM, Logistic Regression (LR), k-NN and Decision Tree (DT); for Study STEM B, the order was the same except for LR and SVM switching places.

ANOVA tests showed statistically significant ($p < .001$) differences between the model accuracies for both datasets, with $F(5,255) = 257.2687$ for Study STEM A and $F(5,255) = 257.4049$, for Study STEM B. The standard deviation was calculated after applying the binomial distribution formula.

The top three models in terms of accuracy, on both datasets, are the same as for predicting Career in STEM (see section 5.1). However, the differences are less pronounced for Study STEM, and the sensitivity and specificity metrics do not show the same clear pattern.

Interestingly, the sensitivity-specificity balances among the top three performers show the opposite relationship from the results on Career in STEM. Overall, the accuracy levels on this task were higher while the balance of nearly all models were worse, mainly due to low levels of specificity; this is not unexpected considering the class-imbalance for the Study STEM target variable, with 70.2% of cases belonging to the positive class, in contrast to the more well-balanced Career in STEM target where the positive class constituted 49.4% of the cases. Only the nB model performed well on specificity for this task, with 70.77% on the Study STEM B data.

Table 2. Results for the five traditional ML algorithms on the Study STEM A and Study STEM B datasets

Algorithm	Accuracy	Sensitivity	Specificity
Logistic Regression	A: 76.74 % B: 76.84 %	A: 88.12 % B: 87.50 %	A: 49.23 % B: 50.77 %
SVM	A: 77.25 % B: 76.82 %	A: 88.12 % B: 87.50 %	A: 50.77 % B: 50.77 %
k-NN	A: 74.11 % B: 73.70 %	A: 85.62 % B: 84.38 %	A: 46.15 % B: 47.69 %
Decision Tree	A: 73.16 % B: 72.35 %	A: 81.88 % B: 78.12 %	A: 50.77 % B: 50.77 %
Naïve Bayes	A: 78.50 % B: 79.88 %	A: 85.62 % B: 83.75 %	A: 61.54 % B: 70.77 %

The results are similar between Study STEM A and Study STEM B, with the exception of the increase in specificity for the nB mode. This, again, highlights the influence that the model selection can have on the outcome. Furthermore, depending on the application of the model, there may also be a preference for better sensitivity or specificity (or indeed a balance between the two) as well as the preference for a simpler model with fewer attributes.

5.3 Deep Learning Epilogue Experiment

Finally, a small epilogue experiment was carried out with a Neural Network model consisting of two layers with 1000 and 8000 Rectified Linear Units (ReLU) [15] respectively, along with a single sigmoid output unit, outputting a value between zero and one; a cut-off point of ≥ 0.5 is used to predict the positive class label. The network was trained with the backpropagation algorithm, using the binary cross-entropy function to calculate the network error.

Each dataset was split into a training and test set, with 66% and 34% using random selection. Training was performed iteratively through repeated exposure of the training set, where one pass through the data is called an epoch. Accuracy was measured on both the training and test set after each epoch.

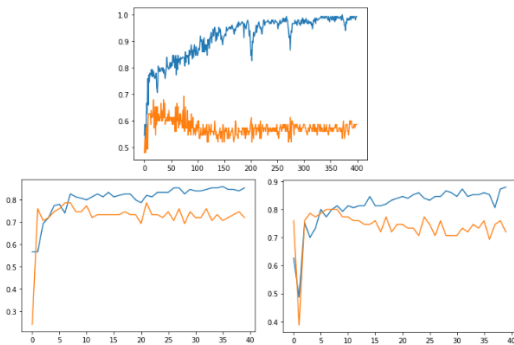


Figure 1. Accuracy at each epoch for Career in STEM (above), Study STEM A (bottom left) and Study STEM B (bottom right). Upper (blue) lines show performance on training data, while bottom (orange) lines show performance on test data.

In Figure 1, the upper lines represent the accuracy on the training set, with the lower lines representing the accuracy on the test set. As is common for this type of model, these accuracies diverge once the network becomes overfitted to the training set. Future experiments will employ techniques for mitigating this, including reducing the number of model weights and implementing early stopping on an additional validation set. The best preliminary results are found at around epoch 80 for the Career in STEM model where the accuracy is near 70% which is roughly on par with the top three traditional algorithms. For the Study STEM, the best test results are found around epochs 8 and 10 respectively, with around 80% accuracy for Study STEM B, and slightly lower on Study STEM A, which again is on par with the traditional ML results.

6. DISCUSSION

The ML algorithms were all able to predict the target variable to some extent. The best accuracies achieved were 72.79% for predicting an interest in a career in STEM and 79.88% for studying STEM. A simple baseline that always predicts the most common class would have accuracies of 50.6% and 70.2% respectively. Thus, while the accuracy for the Study STEM task was higher, the improvement over baseline performance was greater for the Career in STEM task; the latter showed an improvement of 22.19 percentage points or a 43.85% relative increase in predictive performance.

The more advanced algorithms performed better than the simpler models, possibly suggesting that the underlying patterns were too complex to be explained by individual attribute contributions. On the other hand, complexity of interpretation increases along with the complexity of the model. To improve the practical usefulness of applying ML models to this problem, future work will consider various interpretation methods, including advanced techniques such as LIME [18] and SHAP [12] that offers a interpretability for the overall results as well as for individual predictions.

7. CONCLUSION AND FUTURE WORK

This pilot study has demonstrated the potential benefit of Machine Learning (ML) algorithms to model young people’s interest in pursuing STEM careers and education. The more advanced techniques (Logistic Regression, Support Vector Machines and naïve Bayes) achieved higher accuracy levels than the simpler ones (k-Nearest Neighbour and Decision Trees), suggesting that the advanced algorithms may have an advantage in modelling the complex interplay between contributing factors. Future work will aim to distil practically useful insights about these relationships by leveraging existing techniques for interpreting the outcome of ML models. Additionally, we intend to apply these methods on datasets from longitudinal studies to predict changes in interest in STEM after introducing STEM-promoting activities and interventions.

We wish to emphasise that our long-term goal is to use ML algorithms to identify underlying factors that influence interest in STEM careers and further education, to address the predicted skill-gap, and to inform strategies towards more equitable access to STEM jobs. The methods presented here, while predictive in nature, are not intended to be used as prescriptive tools to encourage or discourage individual students of partaking in the STEM field. Even well-intended applications of such nature would need to carefully consider potential consequences, to avoid furthering existing biases and inequities.

8. REFERENCES

[1] Alan, B., Zengin, F.K. and Keçeci, G. 2019. Using STEM Applications for Supporting Integrated Teaching Knowledge of

- Pre-Service Science Teachers. *Journal of Baltic Science Education*. 18, 2 (2019), 158–170.
- [2] Barbara G. Tabachnick and Linda S. Fidell 2001. *Using Multivariate Statistics*. Allyn and Bacon.
- [3] Carnevale, A.P., Smith, N. and Melton, M. 2011. *STEM: Science Technology Engineering Mathematics*. Georgetown University Center on Education and the Workforce.
- [4] Crisp, G., Nora, A. and Taggart, A. 2009. Student Characteristics, Pre-College, College, and Environmental Factors as Predictors of Majoring in and Earning a STEM Degree: An Analysis of Students Attending a Hispanic Serving Institution. *American Educational Research Journal*. 46, 4 (Dec. 2009), 924–942. <https://doi.org/10.3102/0002831209349460>.
- [5] Department of Further and Higher Education, Research, Innovation and Science 2020. *STEM Education Policy Statement 2017–2026*.
- [6] Ergün, A. 2019. Identification of the Interest of Turkish Middle-School Students in STEM Careers: Gender and Grade Level Differences. *Journal of Baltic Science Education*. 18, 1 (2019), 90–104. <https://doi.org/10.33225/jbse/19.18.90>.
- [7] Fadzil, H.M., Saat, R.M., Awang, K. and Adli, D.S.H. 2019. Students' Perception of Learning STEM-Related Subjects through Scientist-Teacher-Student Partnership (STSP). *Journal of Baltic Science Education*. 18, 4 (2019), 537–548.
- [8] Ghent, J. 2005. *A Computational Model of Facial Expression*. National University of Ireland Maynooth.
- [9] Gruca, J.M., Ethington, C.A. and Pascarella, E.T. 1988. Inter-generational effects of college graduation on career sex atypicality in women. *Research in Higher Education*. 29, 2 (Oct. 1988), 99–124. <https://doi.org/10.1007/BF00992281>.
- [10] Ian Witten and Eibe Frank 2005. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann.
- [11] Lent, R.W., Brown, S.D. and Hackett, G. 1994. Toward a Unifying Social Cognitive Theory of Career and Academic Interest, Choice, and Performance. *Journal of Vocational Behavior*. 45, 1 (Aug. 1994), 79–122. <https://doi.org/10.1006/jvbe.1994.1027>.
- [12] Lundberg, S.M. and Lee, S.-I. 2017. A unified approach to interpreting model predictions. *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Red Hook, NY, USA, Dec. 2017), 4768–4777.
- [13] Michie, D., Spiegelhalter, D.J., Taylor, C.C. and Campbell, J. eds. 1995. *Machine learning, neural and statistical classification*. Ellis Horwood.
- [14] Mitchell, T.M. 1997. *Machine Learning*. McGraw-Hill Education.
- [15] Nair, V. and Hinton, G.E. 2010. Rectified linear units improve restricted boltzmann machines. *Proceedings of the 27th International Conference on Machine Learning (ICML-10)* (2010), 807–814.
- [16] Platt, J. 1998. Fast Training of Support Vector Machines Using Sequential Minimal Optimization. (Jan. 1998).
- [17] Quille, K. and Bergin, S. 2019. CS1: how will they do? How can we help? A decade of research and practice. *Computer Science Education*. 29, 2–3 (Jul. 2019), 254–282. <https://doi.org/10.1080/08993408.2019.1612679>.
- [18] Ribeiro, M.T., Singh, S. and Guestrin, C. 2016. “Why Should I Trust You?”: Explaining the Predictions of Any Classifier. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (New York, NY, USA, Aug. 2016), 1135–1144.
- [19] Schölkopf, B. and Smola, A.J. 2002. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press.
- [20] van Tuijl, C. and van der Molen, J.H.W. 2016. Study choice and career development in STEM fields: an overview and integration of the research. *International Journal of Technology and Design Education*. 26, 2 (May 2016), 159–183. <https://doi.org/10.1007/s10798-015-9308-1>.

APPENDIX

Table 3 lists the relevant survey questions that were used in this study. The checkbox answers were translated into attributes with a value of 0 or 1; an additional attribute was created to indicate if no option was selected.

Table 3. Survey questions for the 36 attributes and the two target variables; N = number, B = binned numbers, T = text, L = Likert scale answers given as Strongly Agree, Agree, Neutral, Disagree, Strongly Disagree, CB = checkbox answers (zero or more allowed)

Question	Type
Age	N
County	T
On average, how long, per day, do you spend using technology?	B
On average, how long, per day, do you spend using technology in school?	B
On average, how long, per day, do you spend using technology at home?	B
Confident to work in groups with other.	L
Confident to be creative.	L
Confident to problem solve.	L
Confident to do science tasks.	L
Confident to do maker tasks.	L
Confident to use technology in schoolwork.	L
Confident to use technology to complete coding tasks.	L
I enjoy using technology.	L
I dislike the challenge of Science Technology Engineering and Maths.	Y/N
I am good at projects involving Science Technology Engineering and Maths.	Y/N

Question	Type
What I learn in Science Technology Engineering and Math has no value to me.	Y/N
I do not understand Science Technology Engineering and Maths.	Y/N
Do you have any experience with coding?	Y/N
Homework in Science Technology Engineering and Maths is easy.	Y/N
I struggle in Science Technology Engineering and Maths classes.	Y/N
I would like to participate in more after-school programs in Science Technology.	Y/N
I am good at using technology and completing coding tasks.	Y/N
I understand what Artificial Intelligence (AI) is.	Y/N
Science Technology Engineering and Math is important.	Y/N
Gender	M/F
At home do you have: Computer Tablet Laptop SmartPhone	CB
I use technology to play computer games.	Y/N
I use technology to watch TV.	Y/N
I use technology to talk to friends.	Y/N
I use technology to learn at home.	Y/N
I use technology to code.	Y/N
Do you have family in STEM?	Y/N
I would like to study STEM in the future.	Y/N
I am interested in a career in STEM.	Y/N

Equitable Ability Estimation in Neurodivergent Student Populations with Zero-Inflated Learner Models

Niall Twomey^{*}
Kidsloop Ltd, UK
niall.twomey@kidsloop.live

Sarah McMullan
auticon, UK; Kidsloop Ltd, UK
sarah.mcmullan@auticon.co.uk

Anat Elhalal
Kidsloop Ltd, UK
anat.elhalal@kidsloop.live

Rafael Poyiadzi
Kidsloop Ltd, UK
rafael.poyiadzi@kidsloop.live

Luis Vaquero
Kidsloop Ltd, UK
luis.vaquero@kidsloop.live

ABSTRACT

At present, the educational data mining community lacks many tools needed for ensuring equitable ability estimation for Neurodivergent (ND) learners. On one hand, most learner models are susceptible to under-estimating ND ability since confounding contexts cannot be held accountable (*e.g.* consider dyslexia and text-heavy assessments), and on the other, few (if any) existing datasets are suited for appraising model and data bias in ND contexts. In this paper we attempt to model the relationships between context (delivery and response types) and performance of ND students with zero-inflated learner models. This approach facilitates simulation of several expected ND behavioural traits, provides equitable ability estimates across all student groups from generated datasets, increases interpretability confidence, and can significantly increase the quality of learning opportunities for ND students. Our approach consistently out-performs baselines in our experiments and can also be applied to many other learner modelling frameworks.

Keywords

Neurodiversity, Zero-Inflated Models, Learner Models, Item Response Theory, Data Simulation

1. INTRODUCTION

In the UK, it is estimated that 15% of the population are ND, having neurological functions that differ from what is considered typical [22]. Neurodiversity covers the range of differences in individual brain function and behavioural traits, regarded as part of normal variation in the human population [37]. Each Neurodivergent Condition (NDC) uniquely affects how information is absorbed, processed, and communicated [30, 4]. Our objective is to adapt Learner Models (LMs) for the individual requirements of a number of NDCs

^{*}Corresponding author.

N. Twomey, S. McMullan, A. Elhalal, R. Poyiadzi, and L. Vaquero. Equitable ability estimation in neurodivergent student populations with zero-inflated learner models. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 571–577, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853000>

in learning environments, focusing specifically on dyslexia, dyscalculia and Sensory Processing Disorder (SPD) (with prevalences of 10%, 6% and 5-15% respectively [7, 36, 9]).

Achievement gaps due to NDCs occur early in life and persist through adolescence into adulthood [8]. In many cases, impeded learning opportunities for ND students result from unsuitable learning contexts or lack of adequate student support rather than intrinsically low student ability [29]. However, as learning begins to move further into the digital space [14, 34], LMs, which are statistical models of student attainment, will use historic performance to estimate student ability. Owing to a legacy of potentially poor learning contexts, the ability of ND students tends to be underestimated by LMs since they are not equipped to distinguish between context- and ability-based explanations of performance. Without deliberate effort, therefore, it is very likely that LMs will become biased and offer inequitable recommendations for ND students. On the other hand, opportunities to quell these achievement gaps before they grow are at hand in smart learning environments if LMs are empowered to reason about alternative explanations of performance.

LM research is highly active in the Educational Data Mining (EDM) community. State-of-the-art approaches include deep neural networks [33, 11, 28], and nonparametric Bayesian methods [15]. We find that the literature is sparse for inclusive LMs applied to ND populations, and we were unable to find many bespoke models or datasets (real or synthetic) even in recent literature reviews [1, 21]. Kohli *et. al.* [16] introduced an approach for identifying dyslexic students based on historic patterns of behaviour and artificial neural networks. Mejia *et. al.* [26] approached the task by estimating learner's cognitive deficit specifically for students with dyslexia or reading difficulties. Ensuring the equity of LM is an important area of research, and learning interfaces can be improved by offering multiple assessment Delivery and Response Type (DRT) [29]. Other works have elaborated further on scores and metrics for ethical and equitable recommendation systems with broad stakeholders, including dyslexic students [25]. Equity is also explored along explainability and interpretability axes. Some classical LMs are readily interpretable and offer intuitive explanations of datasets [31, 24], though caution must be exercised to avoid over-interpreting models [13].

ND students face at least two additional hurdles than Neurotypical (NT) students in learning environments: 1) their ability is inaccurately modelled due to LMs shortcomings; and 2) choosing the most suitable learning context for them to express their true ability is rarely considered. Furthermore, the EDM community currently lacks datasets and simulation tools for developing LMs and assessing equity for NDC contexts. We address these three limitations in this work, by motivating and defining equitable LMs for ND students (Sec 2), defining a simulation environment (Sec 2.2), and demonstrating strong performance in our results and conclusions (Secs 3 and 4).

2. METHODS

Due to a lack of available datasets that include ND students, we explore equitable estimation in simulations. Our model combines the use of Zero-Inflated Models (ZIMs) [17] and Item Response Theory (IRT) [2, 20]. Our assumption is that DRT choices will affect the quality of learning opportunities for ND students, with unsuitable DRT resulting in lower Learning Quality Factor (LQF). Without considering the suitability of DRTs for students, LMs risk recommending low-quality learning opportunities and mis-interpreting poor performance on these as an indication of low student ability. The model and simulation procedure proposed is designed to be used to identify the best DRTs for each student, and prevent underestimation of abilities.

2.1 IRT-based Zero-Inflated Learner Model

Our proposed approach, Zero-inflated Learner Models (ZILMs), shown in Eqn (1), builds on the assumption that there are two explicit explanations of zeros: 1) low ability relative to difficulty (low p); and 2) low LQF (high π). With this formulation, a zero from a student with high ability with in an unsuitable DRT can be explained by the poor LQF since π has high responsibility for the outcome [3].

$$Pr(Y = y) = \begin{cases} \pi + (1 - \pi) \cdot (1 - p) & \text{if } y = 0 \\ (1 - \pi) \cdot p & \text{if } y = 1 \end{cases} \quad (1)$$

In our setting, p is based on IRT, and π (which reflects LQFs) is parameterised by item, NDC and DRT features (*c.f.* Sec 2.2), resulting in IRT-based ZILM (IRT-ZILM).

IRT was chosen as the base LM in IRT-ZILM over alternative options as: 1) IRT is well-understood and simple to interpret; 2) Bayesian Knowledge Tracing (BKT) is known to have over- and under-estimation problems [6, 18] that may muddle our understanding of equity for ND students; 3) several technical hurdles need to be overcome to incorporate our approach into BKT; and 4) although Deep Knowledge Tracing (DKT) [33] models can probably learn latent representations that correlate to DRT preferences, this is at the expense of control and interpretation of the effects.

2.2 Simulations

In the simulated dataset, we assume that the ability of ND and NT students are drawn from the same distribution, meaning that ability and NDCs are independent. The NDCs considered in this initial work are dyslexia, dyscalculia, and SPD. These chosen conditions reflect a wide range of effects from different delivery and response types, but this work could be applied to others.

Table 1: Description of parameter distributions used to generate synthetic dataset. Each parameter was randomly assigned from distributions. Users are given an intrinsic ability and the possibility of one or more ND conditions. Items are assigned a difficulty, discrimination, guessing, subject, content type, information density, delivery type and response type. Information density describes how much information is provided—0.1 represents only a few words, 1 is a large block of text—designed to reflect how clearly an item is presented.

Parameter	Value (Range)	Probability
Ability	$(-\infty, \infty)$	$\mathcal{N}(0, 1)$
ND condition	Dyslexia, Dyscalculia, SPD	0.1, 0.06, 0.11
Difficulty	$(-2, 2)$	uniform
Discrimination	$(0.5, 4)$	uniform
Guessing	$(0, 0.15)$	uniform
Subject	Maths, English	0.5, 0.5
Content type	Letter, Digit, Both	M: 0.1, 0.5, 0.6, E: 1, 0, 0
No. attempts	20	fixed
Info. density	$(0.1, 1)$	$\mathcal{N}(0.35, 0.15)$
Delivery type	Read, Listen, Both	0.3, 0.3, 0.4
Response type	Written, Speak, Click Picture,	0.4, 0.2, 0.2, 0.2
	Click Read	

Datasets are created based on the parameters outlined in Table 1. These features contribute to the estimation of LQFs and the probability a user will respond to an item. For example, a dyslexic user’s learning quality is impacted by delivery types involving reading letters, and response types involving reading letters to click the correct answer(s) or writing an answer that includes letters. A dyscalculic user is affected by delivery and response types involving digits. And someone with SPD is impacted when the delivery involves both reading and listening with either letters and/or digits, as this can cause sensory overload [29].

Collectively, these features are used to describe the suitability of DRT to a variety of NDCs, which we now relate back to Eqn (1). If a poorly chosen DRT is selected for a ND student, this will result in poor learning opportunities due

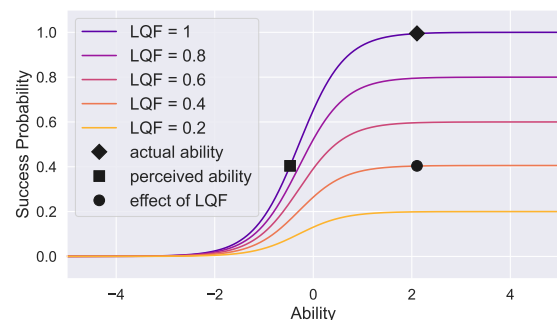


Figure 1: This image shows that LQFs can make the perceived ability of an affected student much lower than their true unobserved ability. \diamond shows a student’s true ability, \circ shows the impact of low LQFs, and \square shows the perceived ability if LQF is not considered.

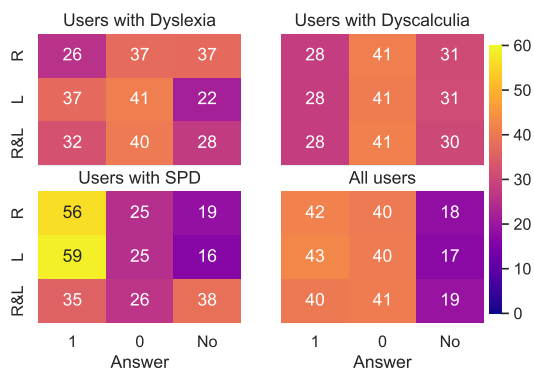


Figure 2: Comparison of attempt outcomes for each NDC (dyslexia: upper left; dyscalculia: upper right; SPD: lower left; all: lower right) when a single delivery type is used for all items (R and L correspond to ‘read’ and ‘listen’ respectively).

to a low LQF (*i.e.* large π). However, if a suitable DRT is selected for a student, the suitability is reflected in higher LQFs. Synthesising datasets that adapt to DRTs and NDCs requires specification of the weight vectors to adapt π to context (*e.g.* ‘reading’ should increase π / reduce LQF for dyslexic but not for dyscalculic students). Although specification of weight vectors is a subjective process, it allows us to express our intuition and instincts about the influential pathways. These are fully described in our implementation¹.

The effect of LQFs on an item’s characteristic curve can be seen in Fig 1. As the LQF decreases, the upper asymptote is reduced, indicating that their opportunity to learn from the interaction is compromised. With this, we interpret LQF as a measure of the contextual inequity.

3. RESULTS AND DISCUSSION

There are four main questions we want to explore in this work: 1) how much are ND users learning opportunities impacted by poor DRTs; 2) is it possible to identify users with potential NDC based on their performance on items with a range of DRTs; 3) is it possible to estimate user true abilities, accounting for any poor performance due to other factors; and 4) can student learning quality and success be improved through active selection of DRTs?

3.1 How are ND users impacted?

Fig 2 shows how ND student performance is affected if a learning environment only delivers information in a single format. Across the full neurodiverse population, the mean performance is approximately the same for all learning material formats. There are also no observable differences in performance for users with dyscalculia. However, for users with dyslexia or SPD there are noticeable differences. For users with dyslexia, they answer 6–11% more attempts correctly and are able to attempt 9–15% more items when the item has a listening component. For users with SPD, they answer an item correctly, and are able to attempt, 19–24% more attempts when the item is only delivered in one format compared to multiple formats. The probability of a

¹github.com/niall-twomey/zero-inflated-learner-models

user succeeding at an item is can be drastically effected by a poor learning quality.

3.2 Can NDCs be identified from interactions?

To investigate if users with a potential NDCs can be identified from the interactions, we have compared individuals mean performance in different subjects and on items with different delivery types (Fig 3). When Maths and English are compared (Fig 3 left), dyscalculic users have attempted more English items than Maths (large spike on ‘Not answered’). Additionally, when Maths is attempted, there is a lower success rate than in English (dip in ‘Correct’). Their performance in terms of ‘Incorrect’ counts in English and Maths are equivalent. However, this tally is achieved with 30% fewer attempts, indicating poor performance in Maths, further illustrating the effect of their NDC (*i.e.* 10/20 *vs.* 5/15). The most noticeable effects between read *vs.* listen DRT (Fig 3 middle) are seen by a clear increase in number of not answered items and decrease in the number of correct answers for ‘dyslexia’ and ‘dyslexia & SPD’ students. SPD students are unaffected by these DRTs. Comparing the ‘read & listen’ and ‘read’ delivery types (Fig 3 right), there are features seen with the dyslexia users, as above, but the SPD users now show a significant difference in performance, with large increases on ‘not answered’ and decreases on ‘correct’. So, by comparing individual students’ performance in different subjects and DRTs, it’s possible to identify the ND students and their condition. In practices, these comparisons could be used to identify what contexts a student may be struggling with, and additional support they may need.

3.3 Can a user’s true ability be estimated?

One aspect of ensuring each user gets suitable learning material is understanding their true ability. Fig 4 compares the performance of classical IRT and our IRT-ZILM model for parameter recovery. With IRT, most of the ability values are under-estimated, particularly for students with 1 or 2 NDCs (Fig 4a). Under-estimated ability makes sense given our expected inflated zero counts. However, the bias of under-estimated ability for ND students is concerning given that ND and NT abilities were drawn from the same distribution. On the other hand, IRT-ZILM is a much better estimator of true abilities (Fig 4b). Additionally, there is no obvious gap in ability estimates for students with NDCs compared to NT students. Table 2 summarises the predictive accuracy of the considered models. Although the performance of all models is approximately equivalent (only small gains for our approach) the lack of distorted recovered parameters may indicate stronger reliability of IRT-ZILM.

Table 3 summarise parameter estimation using Pearson and Spearman correlation coefficients, and have included linear KTM [39] (using contextual features) as another baseline. KTM, like IRT also under-estimates ND ability, and IRT-ZILM is a significantly better estimator of the true parameters.

3.4 Can learning quality be improved?

We explore the effect of actively selecting DRTs to improve LQFs and the number of successful learning attempts for ND students in Table 4. The table shows the potential that selecting the most suitable DRT can have on learning quality, with large lifts on students with 1 or 2 NDCs.

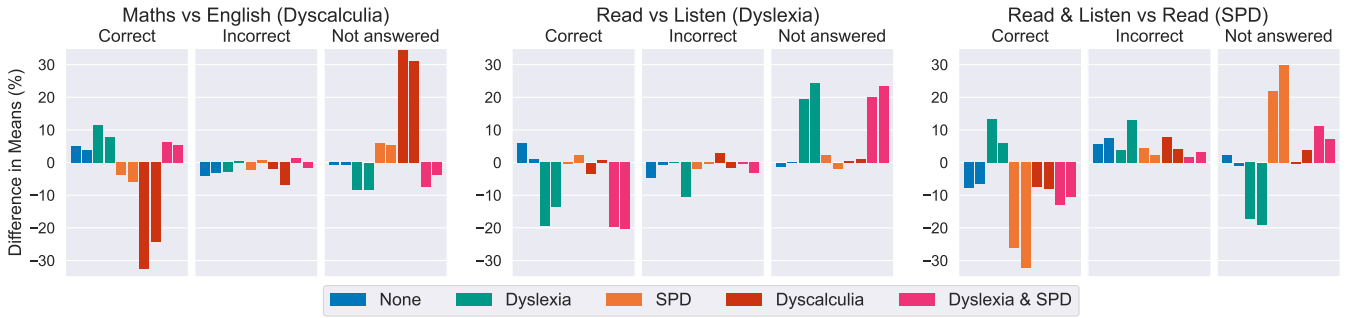


Figure 3: Performance differences for selected students across subject- and NDC-oriented contexts. Bar chart colour indicates NDCs. Large positive and negative values in the bar charts indicates that a group has been affected by the context. While every NDC is affected (indicated in parentheses in subfigure captions), no significant effects present for NT students.

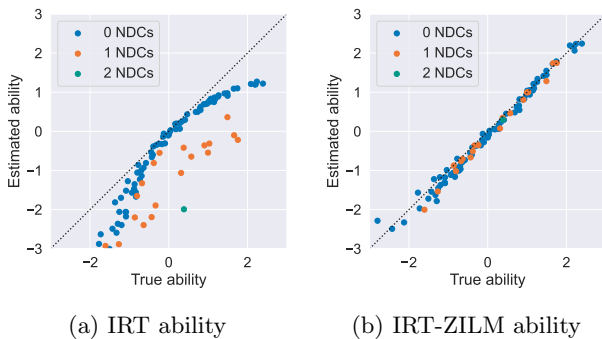


Figure 4: Scatter plots of true vs. estimated ability parameters from IRT and IRT-ZILM. Perfect estimation will place all points on diagonal. IRT is biased against ND students, while IRT-ZILM parameter estimation is very reliable.

Table 2: Predictive test metrics. Similar performance obtained with all models, though IRT-ZILM is slightly more performant than baselines.

Metric	IRT	KTM	IRT-ZILM
Accuracy	0.734	0.742	0.753
F1	0.559	0.567	0.583
NLL	0.513	0.499	0.494
Brier Score	0.170	0.166	0.163

3.5 How can this model be applied?

As already discussed, comparing user interactions in different contexts can identify students who may need additional support in specific areas. Often, high achieving ND students needs can be overlooked since their performance doesn't tend to require interventions. With IRT-ZILM, support/adaptions can be put in place early to enable them to reach their full potential since this model is less susceptible to the biases of traditional LMs. IRT-ZILM can be used to better estimate a students true ability, by adapting it to contexts and underestimating their DRTs preferences. This can help identify and explain causes for underperforming students. By understanding which DRTs a student struggles to engage with, alternative items can be provided to

Table 3: Pearson and Spearman correlation coefficients between true and recovered parameters. Values of 1 indicate perfect matches. IRT-ZILM parameter estimation is the most accurate across both metrics for all parameters.

	Pearson			Spearman		
	IRT	KTM	IRT-ZILM	IRT	KTM	IRT-ZILM
Ab	0.839	0.955	0.993	0.929	0.966	0.996
Diff	0.394	0.686	0.953	0.413	0.707	0.954
Disc	0.270	0.544	0.932	0.234	0.610	0.942

Table 4: Increase (and decrease) of learning opportunities obtained with active (and adversarial) DRT selection.

	1 NDC	2 NDCs
Baseline	0.391	0.123
Lift	1.432 ↑	1.898 ↑
Drop	0.248 ↓	0.014 ↓

help them reach their full potential. These insights can also be used by teachers to explore if the DRTs of their content can be expanded to create an accessible learning environment for all. Education traditionally has taken a one size fits all approach. By harnessing models that incorporate contextual understanding, learning can be tailored to each student, reaching many of those who may previously have felt dejected in learning, as their needs weren't being met.

4. CONCLUSIONS

Our application of zero-inflated models in learning contexts offers a rich simulation environment of neurodivergent conditions in question answering settings, unbiased evaluations of neurodivergent learners, encourages increased learning quality, and more reliably recovers unbiased ability parameters. On the basis of our successful results we believe that further study and exploration of zero-inflated learner models can yield an inclusive framework for equitable, explainable, and reliable learner models in diverse educational data mining contexts. Future work will expand on the experimentation to new contexts, and the model to new domains.

References

- [1] A. Abyaa, M. Khalidi Idrissi, and S. Bennani. Learner modelling: systematic review of the literature from the last 5 years. *Educational Technology Research and Development*, 67(5):1105–1143, 2019.
- [2] M. A. Barton and F. M. Lord. An upper asymptote for the three-parameter logistic item-response model*. *ETS Research Report Series*, 1981(1):i–8, 1981.
- [3] C. M. Bishop and N. M. Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.
- [4] L. E. Boyd, K. Day, N. Stewart, K. Abdo, K. Lamkin, and E. Linstead. Leveling the playing field: Supporting neurodiversity via virtual realities. *Technology & Innovation*, 20(1-2):105–116, 2018.
- [5] Y. Chen, X. Li, J. Liu, and Z. Ying. Item response theory—a statistical framework for educational and psychological measurement. *arXiv preprint arXiv:2108.08604*, 2021.
- [6] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4(4):253–278, 1994.
- [7] J. Crisfield. *The Dyslexia Handbook, 1995*. British Dyslexia Association, 1995.
- [8] E. Ferrer, B. A. Shaywitz, J. M. Holahan, K. E. Marchione, R. Michaels, and S. E. Shaywitz. Achievement gap in reading is present as early as first grade and persists through adolescence. *The Journal of pediatrics*, 167(5):1121–1125, 2015.
- [9] A. Galiana-Simal, M. Vela-Romero, V. M. Romero-Vela, N. Oliver-Tercero, V. García-Olmo, P. J. Benito-Castellanos, V. Muñoz-Martinez, and L. Beato-Fernandez. Sensory processing disorder: Key points of a frequent alteration in neurodevelopmental disorders. *Cogent Medicine*, 7(1):1736829, 2020.
- [10] A. Gelman and J. Hill. *Data analysis using regression and multilevel/hierarchical models*. Cambridge university press, 2006.
- [11] T. Gervet, K. Koedinger, J. Schneider, T. Mitchell, et al. When is deep learning the best approach to knowledge tracing? *Journal of Educational Data Mining*, 12(3):31–54, 2020.
- [12] J. A. Greene, L.-J. Costa, and K. Dellinger. Analysis of self-regulated learning processing using statistical models for count data. *Metacognition and Learning*, 6(3):275–301, 2011.
- [13] K. Holstein and S. Doroudi. Equity and artificial intelligence in education: Will” aided” amplify or alleviate inequities in education? *arXiv preprint arXiv:2104.12920*, 2021.
- [14] B. D. Homer and J. L. Plass. Using multiple data streams in executive function training games to optimize outcomes for neurodiverse populations. In *International Conference on Human-Computer Interaction*, pages 281–292. Springer, 2021.
- [15] M. Khajah, R. V. Lindsey, and M. C. Mozer. How deep is knowledge tracing? *arXiv preprint arXiv:1604.02416*, 2016.
- [16] M. Kohli and T. Prasad. Identifying dyslexic students by using artificial neural networks. In *Proceedings of the world congress on engineering*, volume 1, pages 1–4, 2010.
- [17] D. Lambert. Zero-inflated poisson regression, with an application to defects in manufacturing. *Technometrics*, 34(1):1–14, 1992.
- [18] J. I. Lee and E. Brunskill. The impact on individualizing student models on necessary practice opportunities. *International Educational Data Mining Society*, 2012.
- [19] C.-S. Li. Identifiability of zero-inflated poisson models. *Brazilian Journal of probability and Statistics*, 26(3):306–312, 2012.
- [20] W.-W. Liao, R.-G. Ho, Y.-C. Yen, and H.-C. Cheng. The Four-Parameter Logistic Item Response Theory Model As a Robust Method of Estimating Ability Despite Aberrant Responses. *Social Behavior and Personality: an international journal*, 40(10):1679–1694, Nov. 2012.
- [21] Q. Liu, S. Shen, Z. Huang, E. Chen, and Y. Zheng. A survey of knowledge tracing. *arXiv preprint arXiv:2105.15106*, 2021.
- [22] A. Lollini. Brain equality: Legal implications of neurodiversity in a comparative perspective. *NYUJ Int’l L. & Pol.*, 51:69, 2018.
- [23] B. E. Magnus and Y. Liu. A zero-inflated box-cox normal unipolar item response model for measuring constructs of psychopathology. *Applied psychological measurement*, 42(7):571–589, 2018.
- [24] V. Mandalapu, J. Gong, and L. Chen. Do we need to go deep? knowledge tracing with big data. *arXiv preprint arXiv:2101.08349*, 2021.
- [25] M. Marras, L. Boratto, G. Ramos, and G. Fenu. Equality of learning opportunity via individual fairness in personalized recommendations. *International Journal of Artificial Intelligence in Education*, pages 1–49, 2021.
- [26] C. Mejia, S. Gomez, L. Mancera, and S. Taveneau. Inclusive learner model for adaptive recommendations in virtual education. In *2017 IEEE 17th International Conference on advanced learning technologies (ICALT)*, pages 79–80. IEEE, 2017.
- [27] A. Menon, B. Van Rooyen, C. S. Ong, and B. Williamson. Learning from corrupted binary labels via class-probability estimation. In *International conference on machine learning*, pages 125–134. PMLR, 2015.
- [28] S. Pandey and G. Karypis. A self-attentive model for knowledge tracing. *arXiv preprint arXiv:1907.06837*, 2019.

- [29] T. Papathoma, R. Ferguson, F. Iniesto, I. Rets, D. Vogiatzis, and V. Murphy. Guidance on how learning at scale can be made more accessible. In *Proceedings of the seventh ACM conference on learning@ Scale*, pages 289–292, 2020.
- [30] A. Patrick. *The Memory and Processing Guide for Neurodiverse Learners: Strategies for Success*. Jessica Kingsley Publishers, 2020.
- [31] R. Pelánek. Bayesian knowledge tracing, logistic models, and beyond: an overview of learner modeling techniques. *User Modeling and User-Adapted Interaction*, 27(3):313–350, 2017.
- [32] M. Perello-Nieto, R. Santos-Rodriguez, D. Garcia-Garcia, and J. Cid-Sueiro. Recycling weak labels for multiclass classification. *Neurocomputing*, 400:206–215, 2020.
- [33] C. Piech, J. Bassen, J. Huang, S. Ganguli, M. Sahami, L. J. Guibas, and J. Sohl-Dickstein. Deep knowledge tracing. *Advances in neural information processing systems*, 28, 2015.
- [34] J. L. Plass and S. Pawar. Toward a taxonomy of adaptivity for learning. *Journal of Research on Technology in Education*, 52(3):275–300, 2020.
- [35] E. S. Roemmele. A flexible zero-inflated poisson regression model. 2019.
- [36] R. S. Shalev, J. Auerbach, O. Manor, and V. Gross-Tsur. Developmental dyscalculia: prevalence and prognosis. *European child & adolescent psychiatry*, 9(2):S58–S64, 2000.
- [37] J. Singer. Why can’t you be normal for once in your life? from a problem with no name to the emergence of a new category of difference. *Disability discourse*, pages 59–70, 1999.
- [38] N. Smits, O. Öğreden, M. Garnier-Villarreal, C. B. Terwee, and R. P. Chalmers. A study of alternative approaches to non-normal latent trait distributions in item response theory models used for health outcome measurement. *Statistical Methods in Medical Research*, 29(4):1030–1048, 2020.
- [39] J.-J. Vie and H. Kashima. Knowledge tracing machines: Factorization machines for knowledge tracing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 750–757, 2019.
- [40] L. Wang. Irt–zip modeling for multivariate zero-inflated count data. *Journal of Educational and Behavioral Statistics*, 35(6):671–692, 2010.
- [41] Z.-H. Zhou. A brief introduction to weakly supervised learning. *National science review*, 5(1):44–53, 2018.

Appendix

This section gives supplementary details of our proposed model, see Sec 2.1.

Delivery and Response Weakening

We adapt learner models for NDCs by taking inspiration from techniques used in Weakly Supervised Machine Learning (WSML) [41]. Our approach is to model the interplay between item DRTs and NDCs. Let a binary random variable be drawn from a Bernoulli distribution, $y \sim \text{Ber}(p)$, and let us assume that a label flipping process acts upon y and this results in observations of the corrupted labels, \tilde{y} . The mixing matrix, M , is defined as follows:

$$\begin{pmatrix} 1 - q_0 & q_1 \\ q_0 & 1 - q_1 \end{pmatrix} = \begin{pmatrix} Pr(\tilde{Y} = 1|Y = 1) & Pr(\tilde{Y} = 1|Y = 0) \\ Pr(\tilde{Y} = 0|Y = 1) & Pr(\tilde{Y} = 0|Y = 0) \end{pmatrix}$$

The q_{ij} variables can be selected using prior knowledge and assumptions on the data distributions [27, 32]. In our setting, we are interested particularly in the contexts when learning of ND students is being sabotaged by the environment, *i.e.* q_0 . We therefore model q_0 (previously introduced as a global parameter) and parameterise it with ND, LQF and interaction features.

IRT-based Zero-Inflated Learner Model

Our IRT-ZILM merges LMs and ZILM as follows:

$$Pr(Y = y | \mathbf{x}) = \begin{cases} \pi(\mathbf{x}_\pi) + (1 - \pi(\mathbf{x}_\pi))(1 - p(\mathbf{x}_p)) & \text{if } y = 0 \\ (1 - \pi(\mathbf{x}_\pi))p(\mathbf{x}_p) & \text{if } y = 1 \end{cases}$$

where π and p from Eqn (1) are now functions leveraging ND/LQF/content features (\mathbf{x}_π) and LM/collaborative features (\mathbf{x}_p).

By separating the functional contribution of confounders (π) and ability (p) in IRT-ZILM, we hope to unambiguously decouple these aspects from each other and improve interpretability and explainability. The model is learnt by gradient descent of negative log likelihood of the training data to optimise all parameters. In WSML it is common to learn in a two-step process, for example, by iteratively fixing and optimising IRT and weak label weights [32].

ZIMs have been used to account for excess zeros in many counting tasks using Poisson and negative binomial models [17, 40, 38, 23], and in learning analytics as statistical counting models in self-regulated learning [12]. An important property of statistical models is identifiability as it allows for the precise estimation of the values of its parameters [10, Sec 4.5]. Parallel theoretical analysis has considered identifiability of the counting model parameters [35] and the mixture components [19]. It is worth noting that IRT also suffers from identifiability problems (*c.f.* [5, p.6] and [10, Sec 14.]) but using priors or regularisation can alleviate these.

As far as we are aware, this is the first work to incorporate ZIM in this manner. Choosing IRT as the base LM in IRT-ZILM over alternative options is motivated for several reasons. Firstly, IRT is well-understood and simple to interpret, and using this model as a platform to demonstrate new properties of equity in this early work carries the same benefits. Secondly, BKT is known to have over- and under-estimation problems [6, 18] which may muddle our under-

standing of equity for ND students. Additionally, several technical hurdles need to be overcome, notably adaptation for contextualised individualisation in mixed graphs. Finally, although DKT [33] models can probably learn latent representations that correlate to DRT preferences, this is at the expense of control and interpretation of the effects.

Extra Results

Fig 5 shows this effect for four user/item pairs. For example, the first student should be 60% (orange) successful on this item, however, their LQF is 0.25 (blue), so their success rate drops to 15% (green). Therefore, LQF can be interpreted as a measure of the contextual inequity in these settings.

Although the purpose of this research is to provide equitable estimates of student ability and to provide enabling technology that selects the most appropriate DRT for students, we note that we may also identify students that need additional support in specific areas by recognising potentially unidentified NDCs. We can approach this by creating two models: let \mathcal{M}_0 be the model for a student’s reported NDC state (the ‘null’ model), and let \mathcal{M}_1 be a model trained on data assuming an alternative NDC state (the ‘alternative’ model). Since we have already shown that metrics and likelihood is improved with IRT-ZILM, a statistical hypothesis test can be performed on both likelihoods to determine whether the null or alternative NDC offers a better explanation of data. We leave further elaboration of this approach as future work since it is outside the scope of our direct objectives.

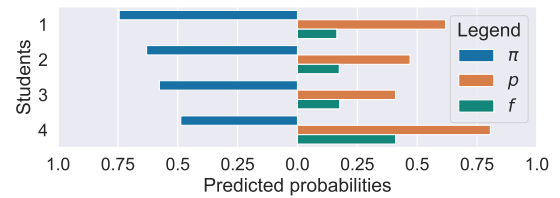


Figure 5: IRT predictions (f) combining with complementary LQFs (π) producing IRT-ZILM predictions (p).

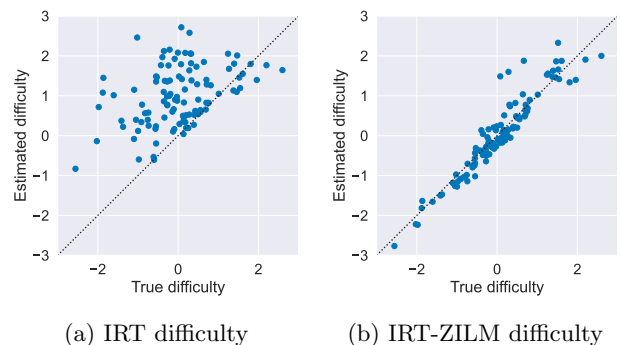


Figure 6: Scatter plots of true *vs.* estimated difficulty parameters from IRT and IRT-ZILM. Perfect estimation will place all points on diagonal. Estimation from IRT-ZILM is significantly more accurate than IRT.

Equity and Fairness of Bayesian Knowledge Tracing

Sebastian Tschatschek
University of Vienna
Faculty of Computer Science
Research Network Data
Science
Vienna, Austria

Maria Knobelsdorf
University of Vienna
Faculty of Computer Science
Computer Science Education
Vienna, Austria

Adish Singla
Max-Planck Institute for
Software Systems
Saarbrücken, Germany

ABSTRACT

We consider the equity and fairness of curricula derived from Knowledge Tracing models. We begin by defining a unifying notion of an equitable tutoring system as a system that achieves maximum possible knowledge in minimal time for each student interacting with it. Realizing perfect equity requires tutoring systems that can provide individualized curricula per student. In particular, we investigate the design of equitable tutoring systems that derive their curricula from Knowledge Tracing models. We first show that the classical Bayesian Knowledge Tracing (BKT) model and their derived curricula can fall short of achieving equitable tutoring. To overcome this issue, we then propose a novel model, Bayesian-Bayesian Knowledge Tracing (B²KT), that naturally allows online individualization. We demonstrate that curricula derived from our model are more effective and equitable than those derived from existing models. Furthermore, we highlight that improving models with a focus on the fairness of next-step predictions can be insufficient to develop equitable tutoring systems.

Keywords

equity & fairness, knowledge tracing, intelligent tutoring

1. INTRODUCTION

In recent years Massive Open Online Courses (MOOCs) and online educational platforms have gained significant importance. They hold the opportunity of providing education at scale and making education accessible to a larger part of the world's population. To facilitate learning in online education and enable customized learning paths for all students, intelligent tutoring systems can be employed while limiting the amount of manual work necessary for each student [11].

In that context, moving education from an offline setting to an online setting, has the potential to promote *Inclusion, Diversity, Equity, and Accessibility (IDEA)*. In particular, by reducing personnel efforts for tutoring, there is the opportunity to include students with diverse backgrounds and

skills, and, importantly, to support their learning equitably. To achieve this, an intelligent tutoring system must be able to adapt to the specific characteristics of each student.

While individualized tutoring has been studied in the community for many years, we consider individualization with a focus on equitable and fair tutoring in this paper. We start by providing a unifying definition of an equitable tutoring system. Our definition is based on the ethical principles of *beneficence* (“do the best”) and *non-maleficence* (“do not harm”) which are commonly adopted in bioethics and medical applications [1]. These principles dictates that we should provide tutoring which maximizes the achieved knowledge while minimizing a student's efforts. In particular we focus on modifying Bayesian Knowledge Tracing (BKT) [2] to better realize these ethical principles. To this end, we propose the Bayesian-Bayesian Knowledge Tracing (B²KT) model and demonstrate its advantages for equitable tutoring in several experiments. Furthermore, we investigate the relation of the commonly considered AUC score concerning the derived tutoring policies, finding that even if a BKT model appears fair in terms of the AUC score, the derived tutoring policies can be inequitable.

In summary, we make the following contributions: *(i)* We propose a unifying definition of equitable tutoring motivated by ethical principles. *(ii)* We propose the B²KT model which allows for effective individualization and demonstrate its benefits concerning equitable tutoring. *(iii)* We highlight that focusing on equity in terms of AUC can be insufficient to ensure equitable tutoring in terms of our definition.

An longer version of this paper with additional experimental results and extended discussion is available [15].

2. RELATED WORK

Fairness in online education and BKT. Several works have considered fairness in data-driven educational systems and intelligent tutoring, e.g., [7, 4, 17, 8]. In [7], the authors discussed implications of using data-driven predictive models for supporting education on fairness. They identified sources of bias and discrimination in “the process of developing and deploying these systems”, and discussed high-level possibilities to improve fairness of systems in the “action step”. In [8, 17], it was investigated how different data sources can provide helpful information to predict students' success in education. Key insights were that different data sources can help to make better predictions but have different characteristics in whether they over- or underestimate students' success [17], and that such predictions can include gender and

S. Tschatschek, M. Knobelsdorf, and A. Singla. Equity and fairness of Bayesian knowledge tracing. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 578–582, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853012>

racial bias in some fairness measures which can be partly alleviated through post-hoc adjustments [8]. In [4] fairness in the context of BKT was studied, and it was found that tutoring policies basing on inaccurate BKT models can be inequitable, when considering the difference in learning success for different subpopulations as a measure of unfairness. Related work also considers adopting a Bayesian perspective for realizing fair decision rules under model uncertainty [3] and fairness in the context of non-i.i.d. data [19].

Individualization in BKT. Several papers have studied individualization of BKT models per student, e.g., [9, 10, 18]. In [10] the *prior per student* model was introduced which uses a student-specific parameter characterizing the students’ individual knowledge. [18] considered individualization through defining student and skill specific parameters which are fitted through gradient descent.

Instructional policies. Key for achieving equity according to our definition are instructional policies which stop practicing a skill at the right time. This problem has for instance been considered in [6, 12]. Further related work has investigated approaches leveraging deep models for creating policies to quickly assess students’ knowledge [16] and using reinforcement learning for optimizing tutoring policies [14, 5].

3. BACKGROUND & NOTATION

Bayesian Knowledge Tracing. Bayesian knowledge tracing (BKT) [2] is a model characterizing the skill acquisition process of students. For a single skill, it can be understood as a standard hidden Markov model in which the binary (latent) state encodes the mastery of the skill, and the binary observations indicate whether a practicing opportunity of the skill was solved correctly. Upon practicing a not yet mastered skill, the student acquires the skill with probability $p(T)$. Once a skill is mastered, it remains mastered. If a student has mastered the skill practiced by an exercise, they solve this exercise correctly with probability $1 - p(S)$. If a student has not mastered the skill, it guesses the correct answer with probability $p(G)$. At the beginning, a student has already mastered the skill with probability $p(L_0)$.

Notation. We consider the interaction of students $s \in \mathcal{S}$ with an intelligent tutoring system. The interaction history up to time t is denoted as $\mathcal{D}_t^s = \{(z_1, c_1), (z_2, c_2), \dots, (z_t, c_t)\}$, where $z_{t'} \in \mathcal{Z}$ is the skill practiced through an exercise at time t' , $c_{t'} \in \{0, 1\}$ is an indicator of whether the exercise was solved correctly, and \mathcal{Z} is the set of skills. In the context of BKT, we refer to the random variables (RVs) indicating whether skill $i \in \mathcal{Z}$ is mastered at time t as Z_t^i and to the RVs indicating whether an exercise practicing that skill would be solved correctly as C_t^i . Sometimes we add another superscript s to indicate the student the RVs correspond to. Upper-case terms like Z_t^i denote RVs and their lower-case counterparts like z_t^i denote particular instantiations.

4. EQUITABLE TUTORING

In this section, we provide a definition of equity in intelligent tutoring and discuss its operationalization.

4.1 Definition

We consider a tutoring setting in which a total of K skills ought to be taught to a set of students \mathcal{S} by an intelligent tutoring system employing a tutoring policy $\pi: \mathcal{H} \rightarrow \mathcal{I} \cup \{\top\}$. This policy maps histories $h \in \mathcal{H}$ consisting of observations

of a student’s learning process to an exercise $e \in \mathcal{I}$ to be practiced next or to a stop-action \top , which ends the teaching process. Each student can have different learning characteristics. Every tutoring policy π has an expected stopping time $T^s(\pi)$, i.e., the expected time of executing the stop action, and an expected knowledge $L^s(\pi)$ acquired by the end of the teaching process, i.e., $L^s(\pi)$ is the expected number of mastered skills upon executing the stop action.

Our notion of equity is based on the ethical principles of *beneficence* and *non-maleficence*. We understand them to translate into the objective of maximizing a student’s knowledge using as little of the student’s resources as possible, i.e., performing a minimal number of exercises:

Definition 1. Consider a tutoring system employing a tutoring policy π . The policy π is equitable for student s iff

$$T^s(\pi) = \min_{\pi', L^s(\pi')=K} T^s(\pi') \quad \text{and} \quad L^s(\pi) = K.$$

A tutoring system is equitable if its tutoring policy is equitable for all students $s \in \mathcal{S}$.

Thus, informally, a tutoring system is equitable if it can teach all K skills in the minimal amount of time possible to any student. Note that our notion of equity is strongly related to that introduced in [4] (cf. discussion below). In the above definition, we implicitly assume that all students can master all K skills.¹ Importantly, a tutoring system can only be equitable if it is adaptive to the students which are interacting with it. In particular, it has to individualize the assignment of exercises and needs to carefully select the “stop action”, in order to achieve equity. The above definition describes an idealized notion of equity which in general cannot be achieved as the tutoring policy would have to teach using the optimal policy right from the beginning. Nevertheless, we can compare tutoring policies π in the spirit of the above definition. In particular, given two tutoring policies π and π' which both teach the same number of skills, we consider the policy π to be *more equitable* as compared to π' if for all students $s \in \mathcal{S}$ it holds that $T^s(\pi) \leq T^s(\pi')$.

We note that our notion of equity is strongly related to that introduced in [4]. In [4], the authors “assume that an equitable outcome is when students from different demographics reach the same level of knowledge after receiving instruction”. The desideratum of achieving knowledge fast is later also added to their notion of equity whereas in our case it is a fundamental constituent. Furthermore, our interest extends to downstream implications of such a definition of equity, namely the individualization of knowledge tracing.

Theoretical Implications. Our definition of equity leads to the following (probably obvious) but important observation:

Observation 1. A tutoring system for a population of students with different learning characteristics can only be equitable if its tutoring policy is adaptive to the students.

Thus, we note that if the tutoring policy is derived deterministically from a non-adaptive, initially incorrect, model of the students, the tutoring system will in general not be equitable. Achieving equity would require basing a policy on rich side information in order to employ an optimal tutoring policy for each student right from the beginning. But such rich side information might not be available.

¹Our definition can be easily generalized to account for an individual student’s maximal achievable knowledge.

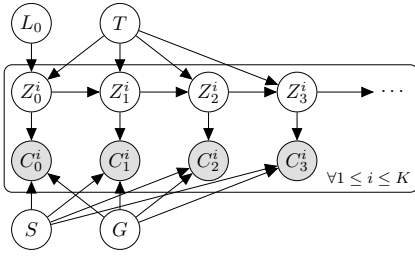


Figure 1: Graphical model of B²KT. The acquisition and application of the K skills depends on $p(L_0)$, $p(S)$, $p(G)$, $p(T)$.

4.2 Operationalization

Tutoring policies are often either simple fixed strategies or derived from a model, e.g., a BKT model, such that each knowledge component is repeatedly exercised until it is mastered with a certain probability. But tutoring policies based on incorrect or non-adaptive models can result in a student not acquiring all skills or suggest to perform too many practicing opportunities. Thus the following two general directions are important for building equitable tutoring systems: (i) **Using side information.** Any available side information about a student should be used to individualize the underlying models. In the context of classical BKT models, the side information could be used to make an initial guess about the key parameters of the model ($p(L_0), p(S), p(G), p(T)$). (ii) **Online adaptation.** Even when using side information, a model is likely not perfectly individualized to all students. To further adjust the models in such cases, online adaption of the models during interaction seems promising.

5. PROPOSED APPROACH: B²KT

In this section, we propose a Bayesian variant of the classical BKT model which enables online adaption to student’s parameters from which individualized — potentially more equitable — policies can be derived, cf. Figure 1.

We assume that each student s has its own learning dynamics, described by student-specific parameters θ^s . If the learning dynamics can be described using a BKT model, $\theta^s = (p(L_0^s), p(T^s), p(S^s), p(G^s))$. We assume these learning dynamics to apply for the acquisition of *all* skills. In practice, we don’t know these parameters and need to infer them. To this end, we take a Bayesian approach, and we assume a set of possible parameters Θ such that $\theta^s \in \Theta$ and a prior distribution $p_0(\theta^s)$. Based on t observations of a student’s practicing exercises collected in \mathcal{D}_t , we can compute the probability that a student has mastered a specific skill and base tutoring policies thereon. As we don’t know θ^s , this requires marginalizing out the (unknown) parameters θ^s . In this way the different possible parameters and their influence for predicting the knowledge state get re-weighted according to the available data. In particular, we compute

$$p(Z_t^{s,i} | \mathcal{D}_t) = \int_{\theta \in \Theta} \underbrace{p(Z_t^{s,i} | \theta, \mathcal{D}_t)}_{=:(\#1)} \underbrace{p(\theta | \mathcal{D}_t)}_{=:(\#2)} d\theta, \quad (1)$$

where $Z_t^{s,i}$ is a random variable indicating whether skill i is mastered at time t by student s . For only a few possible parameters θ , the above equation can be solved exactly by enumeration and by observing that both terms (#1) and

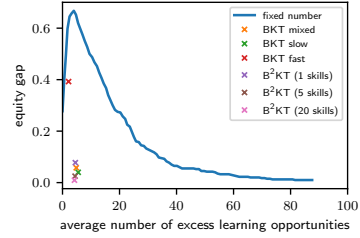


Figure 2: Equity gap vs number of excess learning opportunities. B²KT becomes more equitable as more skills are taught.

(#2) can be computed efficiently by the following recursion:

$$\begin{aligned} \alpha_0^\theta(l) &= p(Z_0^{s,i} = l | \theta) = p(L_0)^l (1 - p(L_0))^{1-l} \\ \alpha_{t+1}^\theta(l) &= p(Z_{t+1}^{s,i} = l, \mathbf{c}_{t+1}^i) \\ &= \sum_{z_t^{s,i}} p(\mathbf{c}_{t+1}^i | Z_{t+1}^{s,i} = l) p(Z_{t+1}^{s,i} = l | Z_t^{s,i} = z_t^{s,i}) \alpha_t(z_t^{s,i}) \end{aligned}$$

Here \mathbf{c}_t^i collects all observations with respect to practicing the i th skill up to time t , and $c_{t'}^i$ is the t' th entry of \mathbf{c}_t^i . Then

$$\begin{aligned} (\#1) &= p(Z_t^{s,i} = 1 | \theta, \mathcal{D}_t) = \frac{\alpha_t^\theta(1)}{\alpha_t^\theta(0) + \alpha_t^\theta(1)}, \text{ and} \\ (\#2) &= p(\Theta = \theta | \mathcal{D}_t) = \frac{p_0(\theta) \cdot (\alpha_t^\theta(0) + \alpha_t^\theta(1))}{\sum_{\theta' \in \Theta} p_0(\theta') \cdot (\alpha_t^{\theta'}(0) + \alpha_t^{\theta'}(1))}. \end{aligned}$$

6. EXPERIMENTS

We perform experiments on synthetic data and consider settings in which the learning rate $p(T^s)$ is assumed to be unknown. This is motivated by previous work which has identified the learning rate as a key parameter for improving BKT based models [18]. In all presented results we denote the average stopping time of a policy for a population of students by T_{stop} and the average number of acquired skills by % skills. We consider THRESHOLD(τ) curricula based on knowledge tracing models. These curricula repeatedly exercise a skill until it is mastered with a probability of at least τ under the model. We consider the following models: (i) BKT: the classical BKT models with fixed parameters; (ii) B²KT: the proposed Bayesian-BKT model.

Students with different learning behaviors. We study the equity of tutoring policies when the students are sampled uniformly from two groups, each containing students with learning dynamics described by a ground truth BKT model. In particular, we build on the experimental setup from [4] where there is a group of slow learners (BKT slow) and fast learners (BKT fast). In [4], the authors also fitted a BKT model to interaction data from students from both groups; we refer to the corresponding BKT model as BKT mixed. The parameters of the considered models are as follows:

	$p(L_0^s)$	$p(S^s)$	$p(G^s)$	$p(T^s)$
BKT slow	0.0	0.2	0.2	0.05
BKT fast	0.0	0.2	0.2	0.3
BKT mixed	0.071	0.203	0.209	0.096

We considered the interaction with 400 students, 200 from the slow and the fast group, respectively, and we compared the performance of THRESHOLD(0.95) tutoring policies based on these models for different numbers of skills that ought to be taught in Table 1. We observe that in the case of mismatch of the student properties and the BKT models used

Table 1: Equity trade-offs of curricula derived from different models/parameterizations.

THRESHOLD(0.95)	1 skill				5 skills				20 skills			
	slow learners		fast learners		slow learners		fast learners		slow learners		fast learners	
	% skills	T_{stop}	% skills	T_{stop}	% skills	T_{stop}	% skills	T_{stop}	% skills	T_{stop}	% skills	T_{stop}
BKT slow	97.00	24.14	99.50	9.49	97.20	122.80	99.90	66.00	97.55	492.64	99.90	183.84
BKT fast	61.00	13.85	97.50	5.96	62.60	71.98	96.10	29.81	64.20	288.76	97.23	120.59
BKT mixed	95.00	23.51	100.00	8.33	95.40	113.67	99.90	40.93	94.53	466.55	99.68	169.86
B ² KT	94.50	24.04	100.00	7.88	97.70	120.87	98.40	32.61	96.68	493.00	96.66	120.05

for the threshold policy, either only a small fraction of the skills (clearly below 95 %) is acquired or that more than necessary time is spent exercising. The mismatch issue is alleviated in the case of the B²KT model (assuming a uniform prior over both types of students), in particular for a larger number of skills. Intuitively this is because, in the case of multiple skills, the model has more opportunities to learn about the students’ characteristics and leverage this knowledge in later tutoring. This fact is also illustrated in Figure 2 in which we reproduce and extend an experiment from [4] in which we compare the “equity gap” (the difference in the percentage of skills mastered by fast and slow students, respectively) to the number of excess learning opportunities. Importantly, B²KT becomes more equitable as more skills are taught.

Out-of-distribution generalization. We test whether B²KT can help with aspects relevant to *inclusion and diversity*. In particular, we consider a stylized mismatch setting in which a tutoring system interacts with students who have a learning behavior not considered when building the system. In addition to the previous two types of students, we assume a third type of learner (BKT med) with the following parameters: $p(L_0^s) = 0.0, p(S^s) = 0.2, p(G^s) = 0.2, p(T^s) = 0.18$. We considered THRESHOLD(0.95) policies based on BKT models of slow and fast learners and the B²KT model with a uniform prior over slow and fast learners. Our results are presented in Table 2. We observe that the performance of the policies derived from the B²KT model have comparable performance to those derived from the true model (although the true model has zero posterior probability) whereas other models yield policies worse in terms of stopping at the right time or teaching the right amount of skills. This property of B²KT can be helpful for promoting inclusion, e.g., when interacting with students who were underrepresented in the data used for building an intelligent tutoring system.

Fair next step predictions do not necessarily imply equitable tutoring. We show empirically that models which might appear to be fair when looking at their AUCs for different groups of students do not necessarily yield equitable tutoring policies. In particular, we again focus on a student population consisting of two groups of students:

	$p(L_0^s)$	$p(S^s)$	$p(G^s)$	$p(T^s)$
Group 1	0.0	0.1	0.4	0.1
Group 2	0.0	0.1	0.2	0.3

We generated data of 400 students (50% from group 1 and group 2, respectively) in a setting with 20 skills and 1000 random exercises from a BKT model. The true model of group 1’s students achieved an AUC of 0.7393 for group 1’s students, while the true model of group 2’s students achieved an AUC of 0.6710 for group 2’s students.

Looking only at the AUC, the two models appear rather inequitable (there is no group parity). Thus it might appear

sensible to aim to use a BKT model for tutoring which has comparable AUCs for both groups in order to promote equity. For instance, a BKT model using parameters $p(L_0) = 0, p(S) = 0.4, p(G) = 0.1, p(T) = 0.65$ achieves an AUC of 0.6719 on group 1’s students and of 0.6733 on group 2’s students, respectively. That is, the AUCs on the two groups are approximately equal. However, when looking at the different models with respect to their tutoring performance using a THRESHOLD(0.95)-policy, we observe a very different picture, cf. Table 3. In particular, the fraction of skills taught differs significantly between the two groups: In group 1 only 28.68% of the skills are acquired by the students on average while in group 2 74.70% of the skills are acquired. This finding is closely related to the observation that models with greatly different characteristics can have similar AUCs [13].

Table 2: Out-of-distribution generalization.

THRESHOLD(0.95)	1 skill		5 skills		20 skills	
	BKT med		BKT med		BKT med	
	% skills	T_{stop}	% skills	T_{stop}	% skills	T_{stop}
BKT slow	99.50	11.28	99.55	56.45	99.59	225.25
BKT fast	90.75	7.61	91.55	37.59	91.70	151.36
BKT mixed	99.50	10.46	99.00	51.71	99.21	211.29
BKT med	98.25	8.82	97.35	45.59	97.84	184.20
B ² KT	98.75	10.33	97.50	48.80	94.19	168.36

Table 3: Fairness in terms of similar AUCs on different groups does not imply fairness in terms of the derived curricula.

group	group fair wrt AUC			true model wrt group		
	AUC	% skills	T_{stop}	AUC	% skills	T_{stop}
group 1	0.6719	28.68	61	0.7393	96.13	308
group 2	0.6733	74.70	64	0.6710	96.35	105

7. CONCLUSION

We considered the equity and fairness of curricula derived from knowledge tracing models, and provided a unifying definition of equitable tutoring systems. Our definition is, in many practical settings, not realizable but suggests that the individualization of tutoring policies to students is key for realizing equity. We proposed the B²KT model, a Bayesian variant of the classical BKT model, and demonstrated in various experiments that it can be beneficial for realizing equitable tutoring systems and promoting IDEA more generally. Furthermore, we highlighted that improving and evaluating models with the main focus on next-step predictions can be insufficient to develop equitable tutoring systems.

8. ACKNOWLEDGMENTS

Adish Singla acknowledges support by the European Research Council (ERC) under the Horizon Europe programme (ERC StG, grant agreement No. 101039090).

Sebastian Tschiatschek acknowledges funding by the Vienna Science and Technology Fund (WWTF) and the City of Vienna through project ICT20-058.

9. REFERENCES

- [1] T. L. Beauchamp, J. F. Childress, et al. *Principles of biomedical ethics*. Oxford University Press, USA, 2001.
- [2] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4(4):253–278, 1994.
- [3] C. Dimitrakakis, Y. Liu, D. C. Parkes, and G. Radanovic. Bayesian fairness. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):509–516, 2019.
- [4] S. Doroudi and E. Brunskill. Fairer but not fair enough on the equitability of knowledge tracing. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge*, pages 335–339, 2019.
- [5] J. He-Yueya and A. Singla. Quizzing policy using reinforcement learning for inferring the student knowledge state. *International Educational Data Mining Society*, 2021.
- [6] T. Käser, S. Klingler, and M. Gross. When to stop? towards universal instructional policies. In *Proceedings of the sixth international conference on learning analytics & knowledge*, pages 289–298, 2016.
- [7] R. F. Kizilcec and H. Lee. Algorithmic fairness in education. *arXiv preprint arXiv:2007.05443*, 2020.
- [8] H. Lee and R. F. Kizilcec. Evaluation of fairness trade-offs in predicting student success. *arXiv preprint arXiv:2007.00088*, 2020.
- [9] J. I. Lee and E. Brunskill. The impact on individualizing student models on necessary practice opportunities. *International Educational Data Mining Society*, 2012.
- [10] Z. A. Pardos and N. T. Heffernan. Modeling individualization in a bayesian networks implementation of knowledge tracing. In *International conference on user modeling, adaptation, and personalization*, pages 255–266. Springer, 2010.
- [11] G. Paviotti, P. G. Rossi, and D. Zarka. Intelligent tutoring systems: an overview. *Pensa Multimedia*, pages 1–176, 2012.
- [12] R. Pelánek. Conceptual issues in mastery criteria: Differentiating uncertainty and degrees of knowledge. In *International Conference on Artificial Intelligence in Education*, pages 450–461. Springer, 2018.
- [13] R. Pelánek. The details matter: methodological nuances in the evaluation of student models. *User Modeling and User-Adapted Interaction*, 28(3):207–235, 2018.
- [14] A. Singla, A. N. Rafferty, G. Radanovic, and N. T. Heffernan. Reinforcement learning for education: Opportunities and challenges. *arXiv preprint arXiv:2107.08828*, 2021.
- [15] S. Tschitschek, M. Knobelsdorf, and A. Singla. Equity and Fairness of Bayesian Knowledge Tracing. *arXiv preprint arXiv:2205.02333*, 2022.
- [16] Z. Wang, S. Tschitschek, S. Woodhead, J. M. Hernández-Lobato, S. Peyton Jones, R. G. Baraniuk, and C. Zhang. Educational question mining at scale: Prediction, analysis and personalization. *AAAI Conference on Artificial Intelligence*, 35(17):15669–15677, 2021.
- [17] R. Yu, Q. Li, C. Fischer, S. Doroudi, and D. Xu. Towards accurate and fair prediction of college success: Evaluating different sources of student data. *International Educational Data Mining Society*, 2020.
- [18] M. V. Yudelson, K. R. Koedinger, and G. J. Gordon. Individualized bayesian knowledge tracing models. In *International Conference on Artificial Intelligence in Education*, pages 171–180. Springer, 2013.
- [19] W. Zhang, J. C. Weiss, S. Zhou, and T. Walsh. Fairness amidst non-iid graph data: A literature review. *arXiv preprint arXiv:2202.07170*, 2022.

Analyzing the Equity of the Brazilian National High School Exam by Validating the Item Response Theory's Invariance

Vitoria Guardieiro¹, Marcos M. Raimundo^{1,2}, Jorge Poco¹

¹Getulio Vargas Foundation; ²Universidade Federal do Rio de Janeiro
vitoriaguardieiro@gmail.com; marcosmrai@gmail.com; jorge.poco@fgv.br

ABSTRACT

Several studies adopt different approaches to examining how economic, racial, and gender circumstances influence student performance in large-scale entrance exams, such as the National High School Exam (ENEM). Using a methodology based on Item Response Theory, ENEM's exam attempts to assess, for each item (question), the curve (function) that relates the participants' abilities to their probabilities of correctly answering the item, which is assumed to hold whichever subgroup, a fundamental premise of IRT called invariance. This work analyzes whether the ENEM 2019 test presented similar curves for subpopulations defined by gender, race, and income, regardless of the participant's actual abilities. Our approach is to analyze the properties of the observed curve for each group and then perform a non-parametric ranking test to compare the equity of each item (question) for each analyzed characteristic. We found that the "Languages and Codes" questions consistently favored male, white, and high-income participants. At the same time, the other three sets of questions (Mathematics, Natural Sciences, and Human Sciences) were considerably more egalitarian.

Keywords

Higher education entrance exams, Grading equity, Item Response Theory, Educational Data Mining

1. INTRODUCTION

The Brazilian National High School Exam (ENEM, for its initials in Portuguese) is one of the most extensive entrance exams globally, having over 5 million participants registered in 2019 [11]. The exam has several functions; on an individual scale, it serves as an admission test to access the federal universities (through the Unified Selection System or SISU) and access to the federal scholarship programs (University for All Program or ProUni). On a collective scale, this exam allows a comparison between schools and municipalities, and it also serves as an indicator for national public

educational policies at the national level [10]. Several studies investigated how sensible characteristics such as income, race, gender, and locality, affect the participants' score [13, 20, 19]. However, most studies use the grade obtained as a direct indicator of the participants' ability without investigating whether exams' grading methodologies are unfairly favoring or disfavoring specific subpopulations.

Since 2009, ENEM's participants' grades have been assigned using Item Response Theory (IRT) methods, which consider the difficulty of each participant's correct questions [7] to assign the grades, in contrast to the Classical Test Theory, where only the number of correct answers matter [4]. IRT creates a probability function that gives, for each question, the probability of a correct answer given the participant's ability. Moreover, the primary assumption of IRT theory is that such function does not vary independently of subgroups of students [7, 16]. This assumption means that, given two groups based on a specific characteristic (*e.g.*, men and women), we expect the proportion of correct answers for a particular question and grade to be similar in both groups. If that is not the case for several questions in a test, the method will end up a sub or super estimating a group's grade, causing inequality between the groups.

We investigate ENEM's 2019 edition to evaluate whether the invariant assumption holds for gender, race, and income level characteristics. For such, we use the assigned grades and participant's answers given by ENEM's IRT evaluation to approximate the Item Characteristic Curves (ICCs)—which are functions of the probability of correct answer given the participant score. We compare the standard deviation of the observed Area Under Curve (AUC) for each group based on a characteristic (such as men and women for the gender). If a group has a bigger AUC than another group, their participants have a more significant overall probability of correctly answering the question, independently of their true abilities. Therefore, questions showing a high AUC standard deviation for a particular characteristic may favor a group. Following the first analysis, we perform a non-parametric ranking test to check if the behavior found in the questions is statistically consistent in the whole test.

Thus, this paper uses statistical analysis to evaluate whether or not the ICC's estimation of IRT theory might favor particular groups. The contributions of our analysis reside in these aspects: (1) Analyzing whether the question estimations *per se* discriminate towards certain groups instead of

V. Guardieiro, M. M. Raimundo, and J. Poco. Analyzing the equity of the Brazilian national high school exam by validating the item response theory's invariance. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 583–587, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852946>

evaluating the grades themselves. (2) Using robust non-parametric statistical tests to determine if these differences are consistently privileging a specific group.

2. DATA

We used ENEM’s most recent microdata from 2019, composed of four objective tests, each containing 45 multiple choice questions and an essay. The tests evaluate the knowledge areas: Mathematics, Languages and Codes, Human Sciences, and Natural Sciences. The Languages and Codes test consists of five foreign language questions, where the participant chooses whether to be evaluated in English or Spanish, and the remaining 40 questions are in Portuguese. Each participant must take all of the tests.

The National Institute of Educational Studies and Research Anísio Teixeira (INEP), responsible for the ENEM, made available the microdata per participant. These data are anonymized and contain the scores obtained, each participant’s answers, registration data (such as city, age, and school), and a non-mandatory socioeconomic questionnaire about (self-declared) race, family income, and parent’s education and profession. We analyze three group characteristics obtained from the questionnaire:

Race: There are five options in addition to “undeclared”: white, black, brown, yellow (Asian), and indigenous. We analyzed only the first three (white, black, and brown), as the others have few participants.

Income: Family income is defined as monthly minimum wages (MW), and the possible range goes from zero to twenty or more MW. Based on Neri et al. (2020) [14], we separate the participants into low (less than 1/2 MW per capita), middle (between 1/2 and 2 MW per capita), and high-income (more than two MW per capita) classes.

Gender: The exam’s registration only allowed for “Female” and “Male” to be mandatory in its selection. Therefore, we do not have information about other gender minorities.

Group	Exam	Foreign Language	
		English	Spanish
White	463,431	307,657	174,199
Black	125,489	65,503	66,069
Pardo	502,996	244,485	281,473
Low Income	660,222	305,442	389,535
Medium Income	390,670	259,454	144,957
High Income	97,881	83,409	16,223
Women	666,905	350,375	347,461
Men	481,868	297,930	203,254
Total	1,148,773	648,305	550,715

Table 1: Number of participants in each group which were regular graduates in 2019 and answered the questions about Race, Gender, and Income Level. Each participant can choose to take the Foreign Language questions in English or Spanish.

As the parameters of the grading methodology used by ENEM are estimated using only the responses of students who are regular graduates in that particular year, our analysis will

use data from such students who answered the questions of gender, race, and income. Table 1 summarizes the number of analyzed participants in each subgroup divided by the different characteristics.

3. ENEM SCORE ESTIMATION

The method used to obtain the ENEM participants’ score is from the Item Response Theory (IRT) [8], which models the probability of a participant responding correctly to an *item* (or *question*) as a function of its parameters and the participant’s *ability* (or *proficiency*). Several increasing monotonic functions are used to model such a relationship such as the Rasch model [15], and the one and two-parameter logistic models [12, 2]. We employed ENEM’s three-parameter logistic function [2], that is the probability of a correct answer by participant j to item i (event $U_{ij} = 1$) given the proficiency parameter θ_j and item parameters a_i , b_i , and c_i :

$$P(U_{ij} = 1 | \theta_j, a_i, b_i, c_i) = c_i + (1 - c_i) \frac{1}{1 + e^{-a_i(\theta_j - b_i)}}, \quad (1)$$

The relationship between $P(U_{ij} = 1 | \theta_j)$ and the parameters a , b , and c is called the *Item Characteristic Curve* (ICC). Figure 1 illustrates an example of this curve.

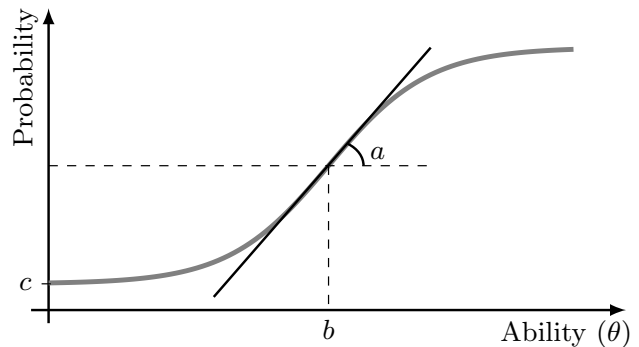


Figure 1: Example of an Item Characteristic Curve (ICC).

Both the item parameters and the participants scores are estimated simultaneously using the participants answers. The scores are estimated, given the item parameters, using the *Expected a Posteriori* (EAP) method with an *a priori* probability function, which is the same for all participants. The *a priori* distribution has mean and variance corresponding to the mean and variance of regular graduating participants in 2009, defined as 500 and 100 points, respectively. More information is available in the participant guides [8, 9] and their bibliographic references [1, 3, 7].

Invariance: Item Response Theory starts from the premise that, for a given question, a single function maps an examinee’s ability to his probability of answering it correctly. Therefore, if the model is well specified, all populations’ parameters are the same. This premise implies the property called *invariance of item and ability parameters*, which is the primary distinction between IRT and classical test theory [7]. Under invariance, we have that the parameters that

characterize an item do not depend on the participants' ability distribution and that the participant ability parameter does not depend on the set of test items.

In practice, invariance does not occur in the strict sense, even when the model is correctly specified [7]. Nevertheless, it is essential to determine the "degree" to which invariance holds. Next, we present how we examine the invariance of the items concerning the subpopulations defined by gender, race, and income.

4. METHODOLOGY

We use the concept of Item Characteristic Curves to compare the performance of each analyzed group in each question. The advantage of such comparison is that it allows us to disregard distinct distributions of scores as we analyze the probability of a correct answer *given* the participant score. For such, we analyze the *observed* ICCs, which were constructed considering the proportion of participants who answered the item correctly for different score ranges.

4.1 AUICC Inspection

Under invariance, we expect the relationship between the participants' scores and the probability of correctly answering a specific question (*i.e.*, the item characteristic curve) to be similar to all groups. Therefore, the area under the item characteristic curve of a specific question should be near equal to all groups. We calculated how different the observed AUICC is for each group of a specific characteristic with the following steps:

1. **Observed ICC:** For each question and subpopulation, the observed ICC was the proportion of participants who answered correctly given their score range.

2. **Item AUICC:** To compare the difficulties of each question, we calculated the area under the item characteristic curve for each subpopulation and the total population. The AUICC can be interpreted as: if a group has a higher AUICC for a question, then it has a higher probability of answering it correctly regardless of possible abilities; therefore, the item is less difficult for that group. Figure 2 shows the Item AUICC calculation.

3. **AUICC discrepancy:** Lastly, we estimated the discrepancy of difficulty for each item and each analyzed characteristic (gender, race, and income) by taking the standard deviation of AUICC for the groups and normalized this value by dividing by the AUICC found for the total population. This normalized value indicates whether inequality seems to hold for this item (small values) or not (bigger values).

4.2 Invariance Checking

The item-by-item AUICC comparison indicates potentially troubling questions. However, to show if there is a consistent difference among the social/gender groups, we performed a non-parametric Friedman ranking test [6, 18] (as implemented in [17]). We performed this test for each exam to see if there are statistical differences among the groups considering a set of items' AUICCs for each social/gender group. For a fixed exam and characteristic, the test procedure involves ranking the groups' AUICC for each question and

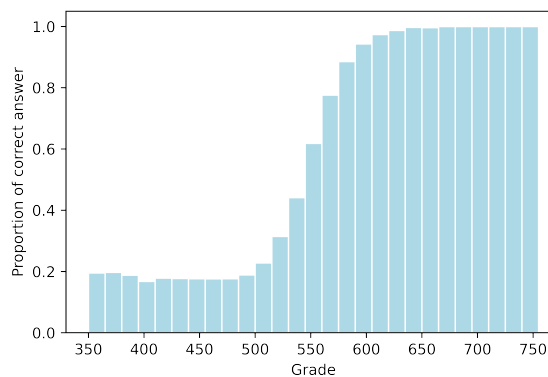


Figure 2: Illustration of the Observed ICC for a given question. The light blue bars represent the proportion of correct answers given by the participants given a grade range. The AUICC is the sum of the area of the bars.

then comparing the ranks obtained for each group. Therefore, if we have n questions and k groups, the Friedman test determines if any of the k groups ranked consistently higher or lower than the others.

For the combination of tests and features whose null hypothesis was rejected by the Friedman test, we determine the one-by-one comparisons through a Finner post-hoc test [5] using the pivot quantities obtained by the previous test. Friedman test checks the hypothesis all groups are equal. If rejected Finner post-hoc test distinguish how each group performs compared to the others.

5. RESULTS

5.1 AUICC Inspection

We performed the Observed ICC analysis to all selected characteristics and the Mathematics (MT), Languages and Codes (LC), Human Sciences (HS), and Natural Sciences (NS) tests. The proportion of corrected answers was calculated for sets of participants' scores with a bin size of 15. For the Languages and Codes test (LC), we separated the questions based on the idiom (Portuguese, English, and Spanish). Every participant takes the Portuguese questions but can choose whether to take English or Spanish as the foreign language questions.

The results are summarized in Figure 3, where for each exam, we have a heatmap where each row of squares indicates which characteristic was analyzed, and the squares in the same columns correspond to the same question. The color of the squares denotes the value of Area discrepancy for that question in the given characteristic, where the darker tones are equivalent to the greater discrepancy, and the symbol indicates which group had the highest area. The color scale is the same for all exams. Therefore, if a particular row contains several dark squares, it indicates that the corresponding exam may be unequal for that given characteristic. If the symbols on the squares are mainly the same, such a given group is more frequently favored in such an exam.

Visually, we can see that a few questions have high diffi-

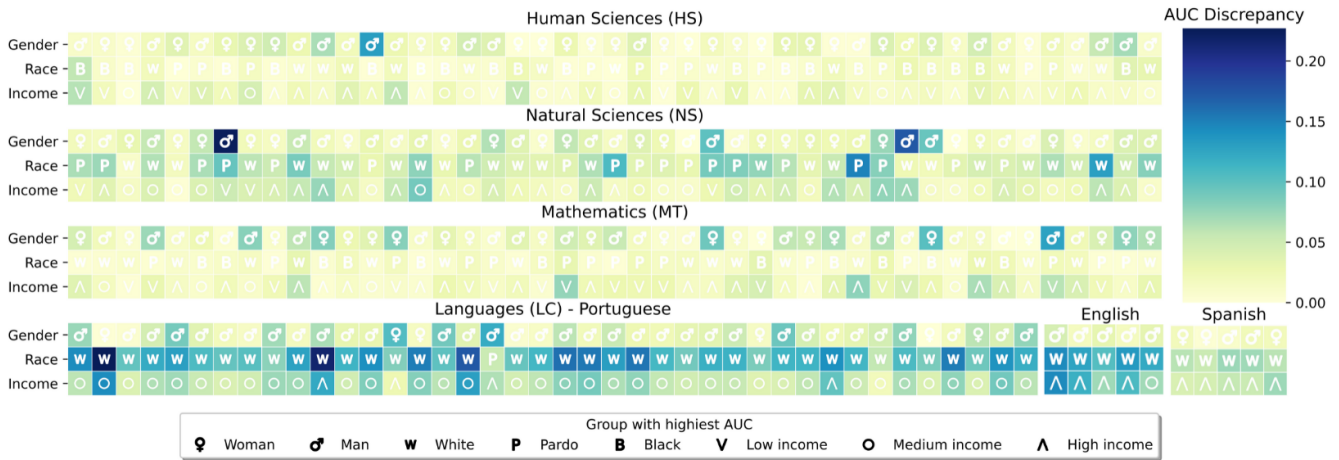


Figure 3: Heatmap of the Area discrepancy based on the Observed ICC. The color of each square indicates the value of Area discrepancy and the symbol indicates the group with the highest area. As the importance of the indicated grows with the discrepancy, the symbols' visibility also grows with the color.

culty discrepancy for the HS, NS, and MT tests. However, the group with the highest area in such questions varies significantly. For instance, in MT and NS, the most discrepant questions are almost equally divided between *man* and *woman*, being the group with the highest area. However, the LC test behaved quite differently. For the race characteristic, almost all of the questions presented high difficulty discrepancy, with the *white* group being dominant as the race with the highest area. Meanwhile, the income groups did not show a consistent dominance of a specific group for the question in Portuguese. However, they did show a strong dominance of the *high* income group in the foreign language questions, mainly in English. This result indicates that the first three tests are egalitarian for gender, race, and income, while the LC test, principally the foreign language questions, is not.

5.2 Invariance Checking

To assess whether the behavior observed in the visual comparison is consistent, we performed the Friedman and Finner tests with the AUICCs. Agreeing with the first experiment, the Human Sciences (HS) and Mathematics (MT) tests did not show a consistent favoring or disfavoring for any group, having a p-value greater than 0.05 in the Friedman test. Meanwhile, the Languages and Codes questions in Portuguese and in English showed a relevant difference among *men* and *women*, with *women* having a lower AUICC rank.

Regarding the Race characteristic, the Natural Sciences (NS) test showed lower AUICC for *black* participants than *white* and *pardo* participants. The Portuguese questions in the LC exam showed similar behavior to the NS test, with the only difference being that the *pardo* participants had a lower AUICC rank than the *white* ones. In both foreign languages questions (English and Spanish), the *black* participants showed a lower AUICC rank than *white*. Lastly, for the Income characteristic, the *low-income* group ranked lower than the *high-income* for NS and also LC in all languages. For the NS test, the *low-income* also ranked lower in AUICC than the *medium-income* participants.

6. CONCLUSIONS

This paper investigated whether the 2019 edition of the Brazilian National High School Exam (ENEM) presented any consistent favor or disfavor for groups based on Gender, Race, and Income Level. Our methodology assessed if the invariance property of Item Response Theory (IRT) holds for Item Characteristic Curves (ICC) estimated for each group given its participants' assigned grades and answers. In our first analysis, we visually compared the overall difficulty of each question and for each group. Then, in our second analysis, we tested if any of the exams were consistently unequal for any group. We found that the Human Sciences and Mathematics questions did not favor or disfavor any group. Meanwhile, the Natural Sciences test was consistently easier for *white* and *pardo* participants in detriment of *black* ones and easier for *high-* and *medium-income* in detriment of *low-income*.

The "Languages and Codes" exam was consistently unequal, with different ICC's (that assigns questions difficulty) behavior for native and foreign language questions. Portuguese questions were overall harder for *women* than *men*, primarily due to the first group having a higher chance of correctly guessing the questions. They were also harder for *black* and *pardo* participants than *white* ones and harder for *low-income* than *medium-* and *high-income*. Foreign language questions also showed inequality, with the English questions favoring *men* in detriment to *women* and both English and Spanish questions favoring *white* participants in detriment of *black* and *high-income* in detriment of *low-income*.

This research also cataloged means of improving IRT: (1) using the analysis of this work to evaluate and possibly reformulate the exam; (2) using data mining methods called multi-task learning to create particular models for each group; (3) using imbalance-robust data mining methods to avoid ICC's bias towards favored groups; and (4) using multi-objective optimization to take into consideration multiple goals (fit the data and keep the model fair). We believe this research contributes to reaching fairer IRT-based tests with the analysis and those future directions.

7. REFERENCES

- [1] F. B. Baker and S.-H. Kim. *Item response theory: Parameter estimation techniques*. CRC Press, 2004.
- [2] A. Birnbaum. Some latent trait models and their use in inferring an examinee's ability. 1968.
- [3] D. F. de Andrade, H. R. Tavares, and R. da Cunha Valle. Teoria da resposta ao item: conceitos e aplicações. *ABE, Sao Paulo*, 2000.
- [4] R. F. DeVellis. Classical test theory. *Medical care*, pages S50–S59, 2006.
- [5] H. Finner. On a monotonicity problem in step-down multiple test procedures. *Journal of the American statistical association*, 88(423):920–923, 1993.
- [6] M. Friedman. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the american statistical association*, 32(200):675–701, 1937.
- [7] R. K. Hambleton, H. Swaminathan, and H. J. Rogers. *Fundamentals of item response theory*, volume 2. Sage, 1991.
- [8] Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira (Inep). *Entenda a sua nota no Enem: guia do participante*, 2012.
- [9] Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira (Inep). *Entenda a sua nota no Enem: guia do participante*, 2021.
- [10] Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira (Inep). Exame nacional do ensino médio (enem) - apresentação, 2022.
- [11] Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira (Inep). Painéis enem, 2022. Last accessed 23 February 2022.
- [12] F. Lord. A theory of test scores. *Psychometric monographs*, 1952.
- [13] R. O. Melo, A. C. d. Freitas, E. d. R. Francisco, and M. T. Motokane. Impact of socioeconomic variables on enem performance: a spatial and sociologic analysis. *Revista de Administração Pública*, 55:1271–1294, 2022.
- [14] M. NERI. Covid, classes econômicas e o caminho do meio: Crônica da crise até agosto de 2020. *Sumário Executivo*, FGV Social, Rio de Janeiro, RJ–Outubro, 2020.
- [15] G. Rasch. *Probabilistic models for some intelligence and attainment tests*. ERIC, 1993.
- [16] S. P. Reise, K. F. Widaman, and R. H. Pugh. Confirmatory factor analysis and item response theory: two approaches for exploring measurement invariance. *Psychological bulletin*, 114(3):552, 1993.
- [17] I. Rodríguez-Fdez, A. Canosa, M. Mucientes, and A. Bugarín. STAC: a web platform for the comparison of algorithms using statistical tests. In *Proceedings of the 2015 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 2015.
- [18] D. J. Sheskin. *Handbook of parametric and nonparametric statistical procedures*. Chapman and Hall/CRC, 2003.
- [19] R. Travitzki, M. E. Ferrão, and A. P. Couto. Desigualdades educacionais e socioeconômicas na população brasileira pré-universitária: Uma visão a partir da análise de dados do enem. *Education Policy Analysis Archives/Archivos Analíticos de Políticas Educativas*, 24:1–32, 2016.
- [20] R. R. Valente. The vicious circle: effects of race and class on university entrance in brazil. *Race ethnicity and education*, 20(6):851–864, 2017.

A deep dive into microphone hardware for recording collaborative group work

Mariah Bradford, Paige Hansen, Ross Beveridge, Nikhil Krishnaswamy, Nathaniel Blanchard
Computer Science Department
Colorado State University
mbrad@colostate.edu

ABSTRACT

Classroom environments are challenging for artificially intelligent agents primarily because classroom noise dilutes the interpretability and usefulness of gathered data. This problem is exacerbated when groups of students participate in collaborative problem solving (CPS). Here, we examine how well six popular microphones capture audio from individual groups. A primary usage of audio data is automatic speech recognition (ASR), therefore we evaluate our recordings by examining the accuracy of downstream ASR using the Google Cloud Platform. We simultaneously captured the audio of all microphones for 11 unique groups of three participants first reading a prepared script, and then participating in a collaborative problem solving exercise. We vary participants, noise conditions, and speech contexts. Transcribed speech was evaluated using word error rate (WER). We find that scripted speech is transcribed with a surprisingly high degree of accuracy across groups (average WER = 0.114, SD = 0.044). However, the CPS task was much more difficult (average WER = 0.570, SD = 0.143). We found most microphones were robust to background noise below a certain threshold, but the AT-Cardioid and ProCon microphones were more robust to higher noise levels. Finally, an analysis of errors revealed that most errors were due to the ASR missing words/phrases, rather than mistranscribing them. We conclude with recommendations based on our observations.

Keywords

Group work, speech recognition, collaborative, spontaneous speech, microphones

M. Bradford, P. Hansen, J. R. Beveridge, N. Krishnaswamy, and N. Blanchard. A deep dive into microphone hardware for recording collaborative group work. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 588–593, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852954>

1. INTRODUCTION

The ubiquity of technology and computers in the classroom has provided unparalleled opportunities to uncover new insights about how people learn in an increasingly digital environment. Successful analysis of learning settings and outcomes — whether through traditional data mining and information extraction, general machine learning, or targeted applications of artificial intelligence (such as intelligent agents in the classroom) — must be able to handle multiple human inputs like structured questions and answers, free-form text, and, importantly, naturalistic human speech that grounds so much of education.

Over the years, various works have assessed the feasibility of utilizing automatic speech recognition (ASR) systems in various environments [1, 4, 6, 7, 8] — of particular note, in 2015, Blanchard et al. [4] evaluated state-of-the-art speech recognition technology on teachers wearing high-quality headset microphones in classrooms, reporting an average word error rate (WER) of 0.44. While [4] found a wide range of ASR performance across major platforms, modern ASR performance is now largely consistent [6]; thus, the primary focus of our work is on which hardware should be used to collect the data to be processed, from real classroom environments. This work highlights budget and performance trade-offs for researchers and practitioners to consider when deploying hardware for studying group work or collaborative problem solving (CPS) [10].

For our evaluation, we simultaneously recorded group audio from multiple microphones. Groups participated in discussion under various conditions that simulate a classroom environment. We then used automatic speech recognition (ASR) to generate separate transcripts from the audio streams of each microphone and used word error rate as our primary metric for evaluating hardware performance. Finally, we took a deep dive into specific errors in the automatic transcription under various conditions to better anticipate how a downstream system might be influenced by said errors. The result is, to our knowledge, a novel assessment of audio sensor hardware for recording collaborative problem solving in a classroom-based data mining system.

2. METHODOLOGY

2.1 Recording setup

We evaluated six microphones: an Audio-Technica ATR2100x-USB (AT-ATR), an Audio-Technica U891Rb Cardioid Condenser Boundary Microphone (AT-Cardioid), an Audio-Technica U891RbO Omnidirectional Condenser Boundary Microphone (AT-Omni), a Blue-Yeti, an MXL AC-404 ProCon (Pro-Con), and a Saramonic SmartMic. Microphones were selected to represent a range of technologies and price points. We assumed that, in a real-world context, one microphone would be used for a single group.

The Blue-Yeti was set to medium-low gain and omnidirectional recording, which are empirically-derived settings motivated by the manufacturer guidelines to maximize the volume of the recording while minimizing clipping. The AT-Cardioid, AT-Omni, and AT-ATR were plugged into a mixer, with the gain set following the same procedure as the Blue-Yeti. The Saramonic was plugged into an auxiliary port (AUX). All other microphones were plugged in with USB. Microphones were all placed on one table five feet from participants. Participant and microphone locations were marked to ensure consistency across groups. Using Adobe Audition, all microphones were synchronously recorded on separate tracks.

Table 1 breaks down our participant demographics. Participants were all students in the Computer Science Department at Colorado State University, all over the age of 18. 26 participants considered English as their first language. Other first languages in the participant pool included Gujarati, Korean, Spanish, Turkish, and Urdu. Providing demographic information was optional and recorded anonymously. Participant groups were recruited by request and all personal information was de-identified.

Table 1: Demographics

Gender	Male	20
	Female	12
	Nonbinary	1
Native Language	English	26
	Non-English	5
	Bilingual	2
Age	18-24	28
	25-31	5

Data was gathered in two types of tests: 1) a prespecified, scripted recording, and 2) a collaborative problem solving task, which are described in Section 2.2 and Section 2.3. In each session, participants sat facing toward the microphone array, and performed both tasks (script first, followed by collaborative task). Each test contained variant noise conditions and was run with 11 ($N = 33$) collaborative triads. Groups were split into two conditions: a “noise” condition where generic “classroom noise” was played at 50% volume from a speaker 10 feet away from the microphone array, and a condition without background noise. The “classroom noise” was pre-recorded classroom sounds including indeterminate chatter. The speaker used was a JBL Bluetooth Flip4 with two 8W amplifiers, corresponding to a maximum 12 dB amplification of the source audio, which was intended

to simulate ambient background noise of an average classroom.

Across all microphones, 7:50:34 total hours of audio were recorded. Specifically, the noise condition constituted 3:40:05 hours of audio, with the remaining 4:10:29 comprising the non-noise condition. Additionally, the scripted task comprised 2:22:38 hours of recording, while the collaborative task constituted 5:27:56 hours of recording.

Each recording from each microphone was separately processed through Google Cloud Automatic Speech Recognition (ASR). Word error rate (WER) was calculated for the full session. The prewritten script served as the ground truth transcript for the scripted portion of the experiment. For the collaborative tasks, transcripts were manually transcribed by researchers, with a lead researcher subsequently verifying the correctness of all the transcripts.

2.2 Group Script Reading Task

In this condition, the participants read a specified script where each participant played a distinct “role” (teacher, student 1, or student 2). The script was a transcription of a real classroom interaction involving two students and a teacher with no overlapping speech.

2.3 Fibonacci Weights Collaborative Problem Solving Task

The Fibonacci Weights exercise is a collaborative problem solving (CPS) group activity where the participants work together to determine the relative weights of five differently colored cubes. The masses of each cube correspond to the Fibonacci sequence. The participants are given a scale, a 10g calibration weight, the cubes, and a worksheet on which to log the weight of each cube when it is determined. The task invites CPS, leading to explicit and implicit coordination, free-form utterances, and overlapping speech.

2.4 Evaluation Metric

We evaluate microphone performance based on ASR performance for word error rate (WER), given by:

$$WER = \frac{S + D + I}{N} = \frac{S + D + I}{H + S + D}$$

where S = the number of substitutions, D = the number of deletions, I = the number of insertions, N = the number of words in the ground truth transcript, and H = the number of successes.

3. RESULTS

3.1 Cross-Task Performance

Figure 1 shows the word error rate (WER) distribution for each microphone across both tasks. A high-level analysis shows that results were relatively consistent across all microphones, with the average WER for every microphone all within 1 standard deviation. The AT-Cardioid showed the best performance, with an average WER of 0.319 ($\sigma = 0.235$). The worst-performing microphone was the Blue-Yeti, with an average WER of 0.365 ($\sigma = 0.278$).

Table 2 provides statistics describing the performance of each microphone across all groups in the script reading task.

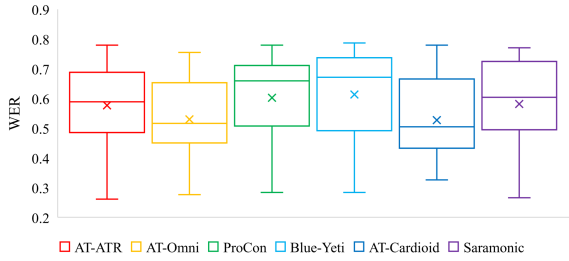


Figure 1: Box-and-whisker plot showing word error rate for each microphone across all groups in the collaborative problem solving task.

Table 2: Descriptive statistics of WER across microphones in the script reading task.

Microphone	μ WER	σ
AT-ATR	0.115	0.051
AT-Cardioid	0.112	0.050
AT-Omni	0.112	0.042
Blue-Yeti	0.117	0.040
ProCon	0.120	0.051
Saramonic	0.107	0.035
All	0.114	0.044

Table 3 provides the same in the collaborative problem solving (Fibonacci Weights) exercise.

In the script reading task, all recordings resulted in a low WER within a very small numerical window of each other, with a low standard deviation across groups. The scripted nature of the task allow for clean turn taking, and a review of the recordings confirm there is little to no overlapping speech during the reading of the script. However, this is not how speech in group work typically manifests. Rather, there are overlaps, incomplete sentences, interruptions, etc. Speakers mumble, speak fast, and rely on implicit communication strategies that are not captured in audio recordings, and therefore on information that is not captured in an automatic speech recognition (ASR) platform’s underlying language model. Put simply, if a typical speech recognition algorithm encounters actual audio as captured in group work, we can expect the WER to be much higher.

Table 3: Descriptive statistics of WER across microphones in the Fibonacci Weights Task.

Microphone	μ WER	σ
AT-ATR	0.577	0.151
AT-Cardioid	0.527	0.137
AT-Omni	0.530	0.128
Blue-Yeti	0.613	0.157
ProCon	0.602	0.150
Saramonic	0.581	0.151
All	0.570	0.143

In the collaborative problem solving task, which contains free-form speech with many overlapping utterances and some disfluencies, we also find that all microphones performed comparably. The clear difference in performance between

the scripted condition and collaborative problem solving condition implies free-form speech (e.g., contains natural interruptions, overlaps, sentence fragments, disfluencies, etc.) was the primary difference in ASR performance. In Section 3.2 we quantify these effects. While the Saramonic was (marginally) the best-performing microphone in the scripted condition, but the AT-Cardioid and AT-Omni produced the lowest mean WER for the Fibonacci Weights recordings.

3.2 Word-Level Analysis

Increased word error rates can be due to three primary factors: *deletions*, where a word in the ground truth transcript is omitted; *insertions*, where an additional word not in the ground truth transcript is added; and *substitutions*, where one word in the ground truth transcript is swapped for a different word.

The majority of errors from the ASR were deletions, while substitution and insertion rates stay relatively stable across both tasks. The most commonly deleted words were relatively constant across microphones. This likely means that the deletions were not due to the quality of the recording, but rather the ASR model itself.

A qualitative analysis of the most frequently deleted words exposes the commonality of such terms in the collaborative problem solving task. For instance, the most commonly deleted word weighted by frequency is the demonstrative pronoun “this”. “This” was dropped 41% of the time and correctly transcribed 53% of the time. In the collaborative problem solving activity we can expect this word to be used frequently, which contributes to the high weighted deletion rate. Likewise, other most commonly deleted words, weighted by frequency, include common particles like “so” or “oh,” and acknowledgments like “yeah” or “okay.” However, some words that in context are important and contentful, such as “10” or “20” (referring to the masses of the weights), are also frequently deleted. See supplemental material for the complete analysis of these commonly missed words.

3.3 Ablation of noise

Classroom environments are by nature noisy [5, 9]. Thus, we ran additional experiments where we increase the noise testing, we had four groups of two participants read the prepared script from Section 2.2 multiple times. With each trial we increased the level of the background noise played by the speaker (a 10% increase was equivalent to a 1.6W increase in speaker power). The net effect of this trial was to control for participants (vocal profile) and ground truth transcript (speech content), while varying background noise level.

Figure 2 shows the results of the noise ablation test. The effect of background noise up to 50% (9 dB) was negligible across all microphones, confirming the results in Section 3.1. When the background noise level is greater than 50% (9 dB), we see the word error rate start to increase significantly with each 10% increase in background noise. With the background noise played at maximum speaker volume, with certain microphones (e.g., Blue-Yeti, Saramonic), we see word error rates start to approach the word error rates demonstrated in the free-form Fibonacci Weight collaborative task, even though the participants in this experiment

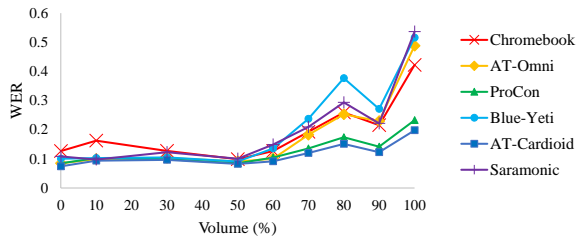


Figure 2: WER vs. background noise volume.

Table 4: Cost of microphones

Microphone	Cost
Chromebook*	\$0.00
Saramonic	\$25.00
AT-ATR	\$99.00
ProCon	\$99.95
Blue-Yeti	\$129.99
AT-Cardioid	\$319.00
AT-Omni	\$319.00

*Baseline microphone

were reading the “well-behaved” prepared script. When the speaker is at maximum volume, this condition should better approximate the noise conditions of a real classroom [5, 9].

3.3.1 Correlation with price

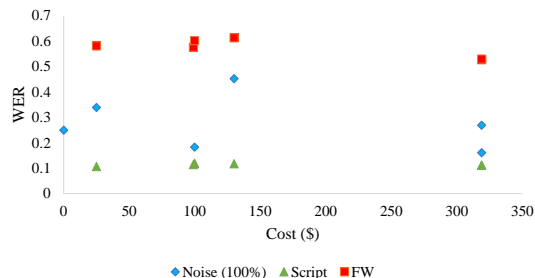


Figure 3: Microphone cost plotted against group mean WER in scripted, Fibonacci weights, and 100% noise ablation conditions.

Finally, Figure 3 shows the mean word error rates across groups in three of the conditions: the script reading, the collaborative task and the noise ablation test session where noise was set to 100%. Together, these three plots show that there is no clear relationship between the cost of the microphone hardware and the ASR performance over associated recordings. Note, the Chromebook served as a “zero dollar” baseline because all other microphones required a laptop to connect to; the microphone integrated with a laptop that students already have access to would entail zero dollars in additional purchases for an in-class recording system.

4. LIMITATIONS AND DISCUSSION

We assessed the quality of a number of microphones by considering how downstream tasks like automatic speech recognition (ASR) performed. At a high level, we found microphones were largely comparable; however, diving deeper, even when the speech is canonically well-behaved the influence of background noise causes dramatic differences in word error rate across microphones at different rates. We found that in the most extreme noise cases, the AT-Cardioid and the ProCon were the most robust to noise, with word error rates (WER) below 0.2 (the no-noise baseline performance was about 0.1). Our overall pick is the ProCon, because it too is highly robust to noise, the price is a third of the AT-Cardioid, and WER is arguably comparable.

Independent of the microphone hardware, off-the-shelf automatic speech recognition performs well (WER less than 0.2) when the recorded speech is canonically well-behaved (e.g., few overlaps, disfluencies, simultaneous speech). However, when participating in group work, there *are* explicit overlaps, disfluencies, and simultaneous speech. Indeed, the presence of these may indicate healthy, productive, educational group dynamics. Still, when recorded speech contains these artifacts, word error rate soars to surprising levels, not dissimilar to the word error rates reported by Blanchard et al. [4] in their study of live teacher speech. Interestingly, they also evaluated scripted and unscripted speech, and found scripted speech error rates were similar to unscripted. Our results showcase that the technology driving automatic speech recognition has improved substantially, since scripted speech is now transcribed with a far lower WER — just not for the kind of naturalistic speech used in collaborative group-work.

There are several limitations of our study. First, we did not verify our conclusions with alternative ASRs (e.g., solutions from Microsoft or Amazon). Several microphones have settings like gain which we hand-tuned, but could be further experimented with. There are, of course, a plethora of microphones we did not include in our test. Finally, we have done our best to replicate classroom environments in a lab setting, but, for now, our evaluation is limited to the lab.

Since most of the word error rate is due to deleted words, a future analysis should consider if it matters if those words are dropped. Taking the transcribed text and evaluating the performance of a further downstream task such as abstract meaning representation (AMR) parsing [2, 3, 11] on it, and on the ground truth transcript, would demonstrate if the dropped words are resulting in significant lost information. However, downstream tasks are likely much more context/project dependent, and being able to predict possible errors in the transcription by understanding limitations of the data gathering process can aid in appropriately designing the downstream tasks by, for example, accounting for the likelihood of missing stop words.

5. ACKNOWLEDGEMENTS

We thank Rosy Southwell, Isaac Courchesne-Owades, and Yongxin Liu for their contributions to data collection and the processing pipeline. This work was supported in part by the National Science Foundation (NSF) under grant number DRL 1559731 to Colorado State University.

6. REFERENCES

- [1] M. H. Asyrofi, F. Thung, D. Lo, and L. Jiang. Crosssar: Efficient differential testing of automatic speech recognition via text-to-speech. In *2020 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pages 640–650. IEEE, 2020.
- [2] L. Banarescu, C. Bonial, S. Cai, M. Georgescu, K. Griffitt, U. Hermjakob, K. Knight, P. Koehn, M. Palmer, and N. Schneider. Abstract meaning representation (amr) 1.0 specification. In *Parsing on Freebase from Question-Answer Pairs. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing. Seattle: ACL*, pages 1533–1544, 2012.
- [3] L. Banarescu, C. Bonial, S. Cai, M. Georgescu, K. Griffitt, U. Hermjakob, K. Knight, P. Koehn, M. Palmer, and N. Schneider. Abstract meaning representation for sembanking. In *Proceedings of the 7th linguistic annotation workshop and interoperability with discourse*, pages 178–186, 2013.
- [4] N. Blanchard, M. Brady, A. M. Olney, M. Glaus, X. Sun, M. Nystrand, B. Samei, S. Kelly, and S. D’Mello. A study of automatic speech recognition in noisy classroom environments for automated dialog analysis. In *International conference on artificial intelligence in education*, pages 23–33. Springer, 2015.
- [5] J. E. Dockrell and B. M. Shield. Acoustical barriers in classrooms: The impact of noise on performance in the classroom. *British Educational Research Journal*, 32(3):509–525, 2006.
- [6] F. Filippidou and L. Moussiades. Benchmarking of IBM, Google and Wit Automatic Speech Recognition Systems. *Artificial Intelligence Applications and Innovations*, 583:73–82, May 2020.
- [7] V. Kėpuska and G. Bohouta. Comparing speech recognition systems (microsoft api, google api and cmu sphinx). *Int. J. Eng. Res. Appl*, 7(03):20–24, 2017.
- [8] F. Morbini, K. Audhkhasi, K. Sagae, R. Artstein, D. Can, P. Georgiou, S. Narayanan, A. Leuski, and D. Traum. Which asr should i choose for my dialogue system? In *Proceedings of the SIGDIAL 2013 Conference*, pages 394–403, 2013.
- [9] B. M. Shield and J. E. Dockrell. The effects of environmental and classroom noise on the academic attainments of primary school children. *The Journal of the Acoustical Society of America*, 123(1):133–144, 2008.
- [10] C. Sun, V. J. Shute, A. Stewart, J. Yonehiro, N. Duran, and S. D’Mello. Towards a generalized competency model of collaborative problem solving. *Computers & Education*, 143:103672, 2020. Publisher: Elsevier.
- [11] C. Wang, N. Xue, and S. Pradhan. A transition-based algorithm for amr parsing. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 366–375, 2015.

APPENDIX

A. WORD-LEVEL ANALYSIS OF ERRORS

Table 5: Top 20 deletions weighted by rate of occurrence

AT-ATR	AT-C	AT-O	Blue-Yeti	ProCon	Saram.
this	this	this	this	this	this
so	so	so	so	the	so
the	the	the	the	so	the
is	is	is	is	is	is
yeah	yeah	yeah	yeah	yeah	yeah
we	i	we	we	we	we
i	we	i	i	i	10
10	one	one	one	10	i
one	10	10	10	one	one
that	that	that	that	that	that
it	it	it	it	it	it
to	okay	okay	to	okay	to
okay	to	to	okay	to	okay
20	a	20	and	and	and
it's	and	a	a	20	a
a	oh	and	20	a	20
and	20	think	oh	be	be
be	think	it's	think	think	like
oh	be	oh	be	it's	it's
think	it's	be	like	oh	oh

Since substitution and insertion rates stay relatively stable across both tasks, it seems clear that in the free-from collaborative problem solving task, the nature of the speech captured in the recordings leads to more deletions. Therefore it becomes more important to understand the nature of the high deletion rates in the collaborative problem solving task. Are there differences in the precise words that are deleted from the recordings made by each microphone? Are these words, when deleted, ones that are likely to have a negative effect on the performance of downstream analysis tasks using the transcription, e.g., parsing or classification?

Table 5 shows the top 20 most frequently deleted words for each microphone, weighted by the overall occurrence of that word in the ground truth, human-generated transcript. Data here was taken from recordings of the collaborative problem solving task only, to analyze the nature of the words being dropped in a group work environment.

Words w are ranked according to $\sum_{t \in T} D_t(w) \times \frac{\sum_{t \in T} C_t(w)}{\sum_{t \in T} N_t}$, where t is a transcript $\in T$ the set of all transcripts, $D_t(w)$ is the number of times w was deleted in transcript t , $C_t(w)$ is the total count of w in the ground truth transcript t , and N_t is the total number of words in the ground truth transcript t . Since this was the free-form activity and each ground truth transcript was different for each group performing the activity, we sum counts over all transcripts.

The most commonly deleted words were relatively constant across microphones. This likely means that the deletions were not due to the quality of the recording, but rather the ASR model itself. In fact, the only words that appear in the top 20 most deleted words that do *not* appear in the top-20 list for every microphone are “it’s,” which did not appear in the top 20 of the Blue-Yeti, “think,” which did not appear in the top 20 of the Saramonic, and “like,” which *only* appeared

in the top 20 of the Saramonic and Blue-Yeti.

A qualitative analysis of the most frequently deleted words exposes the commonality of such terms in the collaborative problem solving task. For instance, the most commonly deleted word weighted by frequency is the demonstrative pronoun “this”. “This” was dropped 41% of the time and correctly transcribed 53% of the time. In the collaborative problem solving activity we can expect this word to be used frequently, which contributes to the high weighted deletion rate. Another commonly deleted word is “one,” a common continuation of “this” as in the bigram “this one,” as might be used to refer to an object in a situated context. Likewise, other most commonly deleted words, weighted by frequency, include common particles like “so” or “oh,” and acknowledgments like “yeah” or “okay.” However, some words that in context are important and contentful, such as “10” or “20” (referring to the masses of the weights), are also frequently deleted.

In reality, most of the deleted words are stop words, indicating that downstream processes should be robust. However, there are some task specific content words that may impede downstream tasks.

Linguistic Profiles of Students Interacting with Conversation-Based Assessment Systems

Carol M. Forsyth, Jesse R. Sparks, Jonathan Steinberg & Laura McCulla
Educational Testing Service
[cforsyth,jsparks,jsteinberg,lmcculla]@ets.org

ABSTRACT

Conversation-based assessment systems allow for students to display evidence of their knowledge during natural language conversations with artificial agents. In the current study, 235 middle-school students from diverse backgrounds interacted with a conversation-based assessment system designed to measure scientific inquiry. There were two versions of the conversations where the initial question was manipulated to examine the relationship between question-framing and conversational discourse. We analyzed the human input during these conversations post-hoc using LIWC to discover linguistic profiles of students that may be related to the type of question asked as well as overall task performance. Furthermore, we compared these linguistic profiles to human ratings as a validity check and to inform our interpretation. Results indicated four separate profiles determined by linguistic features that indeed align to human scores and performance in directions consistent with the effects of question framing. These results offer important implications for improved detection of types of student learners based on linguistic features that do not differ by diverse student characteristics and for designing conversation-based assessments.

Keywords

Conversation-based Assessment, Computational Linguistics, Artificial Agents, Question-Framing

1. INTRODUCTION

1.1 Conversation-Based Assessment

Conversation-based assessments (CBAs) are interactive formative assessment systems with natural language conversations between humans and two or more artificial agents designed to capture evidence of a student's knowledge, skills, and abilities (KSAs) [36]. CBAs leverage previous research on digital learning environments with similar artificial agents or "talking heads" [5, 10, 19, 20, 23,35], artificial intelligence and technology enhanced assessments [4, 6, 26] to create environments where students take actions, answer questions, and participate in conversations to display their

C. Forsyth, J. Sparks, J. Steinberg, and L. McCulla. Linguistic profiles of students interacting with conversation-based assessment systems. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 594–600, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852980>

KSAs. CBAs include other components of simulated, scenario-based, and game-based environments [30, 31, 35] in addition to conversations.

Learning environments with natural language conversations have aided increasing student motivation and deep learning as students converse with artificial agents [1, 2, 21]. Several types of adaptive conversations have been created to accomplish this goal [17, 18] but we will focus on AutoTutor [13] as it greatly influenced the creation of CBAs. AutoTutor is an Intelligent Tutoring System where students have natural language tutorial conversations with an artificial agent; this system shows learning gains comparable to expert tutors in dozens of experiments across multiple domains [13]. Perhaps the "secret sauce" is the adaptive scaffolding moves, which are based on extensive analysis of expert tutor and student interactions [13,15] which include providing hints or broad clues, prompts requiring single word or phrase answers, and assertions as part of the scaffolding framework with associated natural language processing (NLP), which is beyond this paper to discuss [see 13].

CBAs augment the original conversational framework while utilizing the associated NLP of AutoTutor to create more constrained conversations for gathering evidence of KSAs which is necessary for assessment, where less information can be given to the student than in a tutorial dialogue. CBA conversations are designed to elicit information from students that may be difficult to elicit via other means such as multiple-choice or open-ended items. An important issue for CBA design is the impact of question framing in eliciting information from students.

1.2 Question Framing

The need to capture evidence of student knowledge has spurred research in question-asking for decades [3, 8, 15, 22]. This research has yielded multiple taxonomies of questions that seek to elicit student knowledge with respect to both mental representations and cognitive processes [12, 14, 27, 28, 29]. For example, Bloom's taxonomy [3] is well-known for capturing depth of understanding [7]. Specifically, multiple-choice questions provide evidence of shallow or factual understanding while open-ended questions require a deeper conceptual understanding to provide sufficient answers.

Formulating main questions to initiate a conversation with an artificial agent that may require multiple turns to elicit evidence from students is quite different from formulating a single question asked with a constructed response item. Therefore, we drew from a detailed taxonomy [15] to consider specific types of questions that may help elicit information and inform cognitive processing [14,

15]. This taxonomy includes 16 question types at varying levels of depth. Within a CBA task, however, constraints including the target constructs, context, and scenario narrow the possibilities of questions that are appropriate for assessment (e.g., comparison and justification questions). Notably, research shows that tasks requiring students to generate justifications for their responses, such as constructing arguments, can sometimes lead to better learning than comprehension tasks [11]. Therefore, we investigated student responses to two separate types of question framing in a CBA task for science inquiry that prompted students to make a comparison between two artifacts (i.e., observation notes collected from simulated data), contrasting an approach where students were asked to *make a selection and explain their choice* (comparison-framing) with one in which students are asked to *justify whether or not they agree with a choice made by a virtual agent* (argumentation-framing) [34]. Previous research suggests that this manipulation had an impact on CBA conversation performance with the argumentation-framing condition making better selections but showing poorer explanations than the comparison-framing condition [33].

Previous research on CBAs suggests that additional computational linguistic analysis beyond the NLP algorithms needed to operate the system can provide fine-grained insights on how students respond [9]. Furthermore, CBAs offer an opportunity to explore methods for inferring information about students' experiences from their responses in ways that may have implications for equity issues in assessment, given the wide range of responses that can be accepted and awarded credit. Thus, in this study we analyzed responses from a diverse sample with a bottom-up approach using linguistic features and associated profiles, contextualized them within question framing conditions, and compared these profiles based on human scoring and on task performance.

2. METHODS

2.1 Participants

In total, 235 middle school students were recruited from one rural (35%) and one urban (65%) school, yielding quite a diverse sample. Specifically, the sample was 48% female and 52% male, 34% free/reduced price lunch eligible, and 79% White, 9% Black/African American, 4% Hispanic/Latino, and 8% other. The experiment was approved by an IRB and all personally identifiable information was removed from the data.

2.2 Tasks and Measures

2.2.1 Conversation-Based Assessment Weather Task

The Weather CBA task [32, 34] is an innovative, computer-based task that engages students in simulated science inquiry around the topic of thunderstorms, including data collection, analysis and prediction, and justification of reasoning in the context of conversations with two virtual agents: Art, a virtual peer working on the task alongside the student, and Dr. Garcia, a scientist and authority figure guiding the students. Students place weather stations to collect simulated data on an impending thunderstorm, take notes from the data, and in a conversation-based item are presented with two of the notes (one of their notes, and one "created" by the virtual peer). Students are asked to explain which note should be kept for making predictions about the likelihood of a thunderstorm. Students should choose the note summarizing data across multiple weather stations (i.e., more data yields a better prediction).

Students were randomly assigned to one of two question framing conditions, which included different main questions posed to the student [34]. In both conditions, students were presented with the two notes and were asked to indicate which note should be kept for

making predictions later. In the comparison-framing condition, the main question is posed as "Please look carefully at and compare these two notes. Which one do you think we should keep for making predictions later and why?" whereas in the alternate argumentation-framing condition, the question is posed as "Please look carefully at and compare these two notes. I think we should keep [your note/my note] for making predictions later. Do you agree with me? Why or why not?" In this argumentation-framing condition, the peer always pointed students to the "better" note summarizing data from multiple stations. Thus, the conversations included adaptivity based on the student's responses to the simulation components and their linguistic input.

2.2.2 Overall Task Performance

The Weather CBA task includes a combination of more traditional item types (multiple choice, constructed response) and technology-enhanced items (drag-and-drop, simulation items), in addition to simulated conversations characteristic of CBAs, with a maximum total score of 29 points.

2.2.3 Human Scores for Conversations

Responses were dichotomously scored by human raters along multiple dimensions (see Table 1), with a maximum of five possible points per conversation. Responses were scored both for the correctness of students' conclusions and the quality of the supporting reasoning, summed to create a single score for the entire response to the conversation (across all conversational turns). Raters were trained by a scoring leader to score these responses using a well-defined rubric. After two raters independently rated each category as 0 or 1, inter-rater reliability was examined. Initial inspection of the data revealed that there was an unequal cell distribution which can skew kappa results, so we only report percent agreement by dimension: Note Choice (93%); Immediate (90%); Relevant (82%); Sufficient (84%); and Aligned with Note Choice (88%).

Table 1. Dimensions of Human Scores and Definitions

Dimension	Definition
Note Choice	Students choose the note with more complete data represented (observations from multiple weather stations).
Reasoning: Immediate	Students provide reasoning immediately, within the first turn of the conversation.
Reasoning: Relevant	Students mention features related to the notes and the data they contain (e.g., weather stations, water vapor, instability).
Reasoning: Sufficient	Students mention that one note has more data than the other note.
Reasoning: Aligned with Note Choice	Students provide reasoning that is consistent with their choice of note.

2.2.4 LIWC

Linguistic Inquiry and Word Count (LIWC) is a computational linguistic tool that primarily uses a bag of words approach which identifies words in a given text and compares it to categories of words corresponding to parts of speech or at times broader constructs such as affect [24]. Overall, there are currently over 90 features that have shown to predict outcomes such as college GPA [25] to relationship longevity [16]. The system is available for research or for real time use, with an available API (<https://www.receptiviti.com/liwc>). In this study, we used a licensed desktop version of LIWC2015 [24] for post-hoc analysis of

the student contributions to the conversations with the artificial agents in the CBA task.

3. ANALYSES AND RESULTS

We began by analyzing all text with LIWC2015. Next, we inspected the data by question framing condition for sparseness (i.e., more than 95% of features with 0's), such that sparse features were removed. We also removed two outliers with an over-abundance of exclamation marks which were unique to these two students only (<1% of data lost). Next, we conducted a k-means cluster analysis on the remaining 34 features for 233 students. The results yielded four unique profiles discovered with a bottom-up approach. We interpreted the clusters via several methods including inspecting final cluster center means across features, qualitative analysis, and relationships to external variables including student demographics, question framing condition, human scoring of the conversations, and overall task performance.

3.1 Profiles

In total, 34 features were entered into a k-means clustering algorithm, which converged after 9 iterations. The results yielded 26 features with significant cluster centers (see Appendix A). The four clusters were titled Shallow Performers, One-Turn Wonders, Low Performers, and High Performers, with each described below.

3.1.1 Shallow Performers

The first profile was entitled “shallow performers” as these students were likely to give the correct answer to the question in terms of note choice (i.e., were above chance at picking the note that included more data) but did not provide good reasoning for their answers. The linguistic profile of these students as indicated in Appendix A included relatively higher levels of emotional tone ($M=67$, $SD=38$), positive emotions ($M=8$, $SD=9$), affect ($M=9$, $SD=9$) and the highest amount of dictionary words ($M=90$, $SD=8$), which is reverse coded indicating more words that are not in the dictionary, as well as a low word count ($M=14$, $SD=8$) compared to the other profiles. Qualitative analysis indicated that this group produced several responses in which students appeared to be implying that the peer agent is “smart” (e.g., argumentation-framing: “i do [agree] because you have good ideas. _ I do, because you have good ideas and you sound smart.”; comparison-framing: “We should keep Art’s. Art’s notes have more vocabulary and are more descriptive_ I don’t know. Maybe because you are smarter.”). This illustrates positive emotional tone with words such as “good” and “smart/smarter”. The argumentation-framing responses earned credit for agreeing with peer’s note choice but little credit for their explanations; comparison-framing responses were just above chance (60%) in note choice and also had quite poor reasoning.

3.1.2 One-Turn Wonders

Overall, these students provided “immediate” reasoning on the first turn of the conversation, but the likelihood of this reasoning being relevant was around chance levels; the reasoning was only rarely sufficient and aligned to note choice, although comparison-framing condition students scored slightly higher. Argumentation-framing condition responses were overwhelmingly likely to make the correct note choice, while comparison-framing responses were at chance levels. As seen in Appendix A, these students had the highest mean level of affect ($M=10$, $SD=8$), positive emotions ($M=10$, $SD=8$) and words per sentence ($M=17$, $SD=11$). From qualitative analysis, we saw that the first response included apparently longer sentences, consistent with providing immediate reasoning on turn one, and often used the word “better” (positive affect). An example argumentation-framing response is “Yes, you have a very good

note. (I guess) _ It has a bunch of good information on it.”. The student includes reasoning in the first turn (i.e., yes followed by attempted justification); with prompting, they provide additional detail (“good information”) but no relevant reasoning. An example comparison-framing response is: “I personally think my notes are better because i did notes for each seperate weather station. _ I had more hours to predict what happen and i wrote seperate notes for each weather station.” This illustrates higher words per sentence and an attempt to provide immediate reasoning, but the note choice and reasoning are entirely incorrect. It appears these students understand the need to provide reasoning to support their note choice, but this reasoning was not relevant or sufficient.

3.1.3 Low Performers

These students performed poorly on all aspects of the task and human scores, with note choice at chance levels. Their linguistic profile as seen in Appendix A, showed a relatively low level of words per sentence ($M=12$, $SD=8$) which could indicate minimal attempts to provide a viable answer. These responses had high levels of personal pronouns ($M=18$, $SD=10$), perhaps an artifact of requiring students to say “my note” when choosing their own note. Qualitative analysis indicated that comparison-framing students often incorrectly chose “my note”, and argumentation-framing students sometimes disagreed with Art. Their reasoning was more relevant than the shallow performers, but it was similarly insufficient and misaligned to note choice (although somewhat better for comparison-framing). This group also had more off-task responses, perhaps indicating disengagement. An example argumentation-framing response is “No because i did not give a water vaper percent _ no because i did not give a vaper number _ i did not give an exact answer”, showing relevant but insufficient reasoning and incorrect note choice. A comparison-framing response with incorrect note choice and poor reasoning is “mine because im an actual person _ because im an actual person and not a computer program”.

3.1.4 High Performers

These students performed relatively well on all human scores. As seen in Appendix A, these students had relatively high levels of words per sentence ($M=15$, $SD=12$) and analytical thinking ($M=43$, $SD=30$) which makes sense given the science inquiry task context. An example response is “yes because my note shows that there are no cold fronts _ There are no cold fronts” which includes the correct note choice, immediate reasoning on the first turn, and relevant (but not sufficient) reasoning (i.e., mentioning cold fronts) aligned with the note choice. Therefore, these students appear more likely to draw on relevant evidence within the task (above chance levels), with comparison-framing students especially likely to provide correct responses with more relevant reasoning.

3.2 Relationships to External Variables

Profiles were compared to external variables. Specifically, we examine how features correlate with question-framing conditions and overall performance (see Appendix B), followed by analyses of the relationship between profiles to demographics. Relationships between profiles and question-framing conditions, human scores, and overall performance, were examined using a Kruskal-Wallis method with Monte Carlo simulation across 10,000 samples.

3.2.1 Profiles and Demographics

We conducted chi-square analyses between demographic variables and the four profiles. We discovered no significant differences leading us to interpret that profile membership was indeed diverse.

3.2.2 Profiles and Question-Framing Condition

There was a significant difference between profile membership and question-framing condition, ($X^2(3,233) = 8.07, p = .04, \text{partial } \eta^2 = .035$). The Monte Carlo simulation for significance with 10,000 samples revealed a significance level of $p = .04$ (lower bound $p = .038$, upper bound $p = .049$). Mean ranks indicate that the Shallow Performers had the highest number of cases in the argumentation-framing condition (130.34) and the High Performers had the lowest (101.60). Although this may simply be an artifact of random assignment, it makes sense as students in the comparison-framing condition performed better overall. The other two profiles of One-Turn Wonders and Low Performers were in the middle with slightly higher mean ranks for the former over the latter (123.91 and 116.36, respectively).

3.2.3 Profiles and Human Scores

3.2.2.1. Note Selection

A significant relationship was discovered between the profiles and human scores for note selection ($X^2(3,233) = 10.106, p = .02, \text{partial } \eta^2 = .046$). The Monte Carlo simulation for significance with 10,000 samples revealed a significance level of $p = .02$ (lower bound $p = .014$, upper bound $p = .020$). The mean ranks suggest the highest scores for the Shallow Performers (129.38) and the lowest for the Low Performers (98.51) with the One-Turn Wonders and High Performers having virtually equivalent mean ranks (121.59 and 121.98, respectively). These results are consistent with the fact that argumentation-framing students were overwhelmingly likely (~90%) to make correct note selections in both the Shallow and Low Performing profiles, vs. 50-70% for comparison-framing.

3.2.2.2. Immediate Reasoning

A non-significant relationship was not discovered between the four profiles and Immediate ($X^2(3,233) = 4.267, p = .234, \text{partial } \eta^2 = .018$). Patterns revealed that One-Turn Wonders had the highest overall mean (126.64) though essentially similar to that for High Performers (122.21), but substantively different from both Shallow Performers (111.73) and Low Performers (108.51).

3.2.2.3. Relevant Reasoning

A significant relationship was discovered between the four profiles and relevant reasoning ($X^2(3,233) = 20.896, p < .001, \text{partial } \eta^2 = .089$). The Monte Carlo simulation for significance with 10,000 samples revealed a significance level of $p < .001$, (lower bound $p = .000$, upper bound $p = .001$). Mean ranks indicate that High Performers performed the best (142.19), whereas the Shallow Performers performed the worst (92.48). Not surprisingly, the One-Turn Wonders and Low Performers fell in the middle, with One-Turn Wonders showing higher scores (122.00 and 107.19, respectively). Relevant reasoning was most likely for High Performers and One-Turn Wonders in the comparison-framing condition.

3.2.2.4. Sufficient Reasoning

A significant relationship was discovered for the four profiles and sufficient reasoning ($X^2(3,233) = 13.974, p = .003, \text{partial } \eta^2 = .061$). The Monte Carlo simulation for significance with 10,000 samples revealed a significance level of $p = .003$, (lower bound $p = .001$, upper bound $p = .004$). Mean ranks were highest for the High Performers (133.71) and lowest for the Shallow Performers (98.63) with One-Turn Wonders having higher scores than the Low Performers (124.38 and 109.05, respectively).

3.2.2.5. Supports Note Choice

A significant relationship was discovered for the four profiles and supporting note choice ($X^2(3,233) = 18.304, p = .001, \text{partial } \eta^2 = .084$). The Monte Carlo simulation for significance with 10,000 samples revealed a significance level of $p < .001$, (lower bound $p = .000$, upper bound $p = .001$). Once again, mean ranks revealed High Performers had the highest score (139.99) and Shallow Performers had the lowest score (103.99). The One-Turn Wonders and Low Performers fell in the middle with higher scores for the One-Turn Wonders (115.21 and 105.44, respectively).

3.2.4 Profiles and Overall Task Performance

A significant relationship was discovered for the four profiles and overall CBA task performance ($X^2(3,233) = 11.332, p = .010, \text{partial } \eta^2 = .048$). The Monte Carlo simulation for significance with 10,000 samples revealed a significance level of $p < .001$, (lower bound $p = .007$, upper bound $p = .012$). Mean ranks indicate the highest score for the High Performers (134.79) and the lowest for the Shallow Performers (98.41) with One-Turn Wonders still outperforming Low Performers (126.94, 106.16, respectively).

3.3 Results and Conclusions

We discovered four profiles of students based on linguistic features identified with the computational linguistic tool LIWC. The resulting profiles consisted of Shallow Performers, One-Turn Wonders, Low Performers, and High Performers. The Shallow Performers had higher levels of emotional tone, especially positive affect words, but fewer dictionary words. The One-Turn Wonders were characterized by high positive affect words and high words per sentence, reflected in their longer attempted justifications for their choices. The Low Performers showed low words per sentence and high levels of personal pronouns, in part reflecting the design of the task, but also reflecting generally shorter responses with poorer quality of reasoning. Finally, the High Performers had relatively high levels of words per sentence and analytical thinking which has predicted GPA in previous studies [25]. These four profiles were then validated by external measures that revealed patterns in expected directions.

Given the IDEA conference theme and our diverse sample, we compared key demographic measures including school location, race/ethnicity, free/reduced price lunch status, and other factors to the four profiles and found no significant differences. This indicates that these linguistic profiles did not show different demographic composition even though these factors did relate to overall task performance in prior research [32]. The implications are that linguistic profiles may be a manner of detection and intervention that transcend demographics and therefore may enable greater inclusion during the learning experience. That said, we do acknowledge that our sample was predominantly White (about 80%) despite the large variation in other key variables. We also acknowledge the relatively small sample size and plan to attempt to replicate these findings on a larger data set from this same CBA task in future work.

In sum, these findings can guide the creation and augmentation of novel CBAs to support personalized learning based on students' interactions with the system, transcending demographic differences. The methodology employed may also inform other researchers' attempts to discover ways to adapt and personalize other learning and assessment environments based on students' use of language.

4. REFERENCES

- [1] Atkinson, R. K. 2002. Optimizing learning from examples using animated pedagogical agents. *Journ. of Educ. Psychology*, 94,2, 412-427.
- [2] Baylor, A. L., and Kim, Y. 2005. Simulating instructional roles through pedagogical agents. *Intern.l Journ. of Art.Intelligence. in Educa.*, 15, 95-115.
- [3] Bloom, B. S. 1956. Taxonomy of educational objectives: The classification of educational goals. Handbook I: Cognitive Domain. New York: McKay.
- [4] Bennett, R. E., Persky, H., Weiss, A., and Jenkins, F. 2007. *Problem-Solving in technology rich environments: A report from the NAEP technology-based assessment project*. NCEs 2007-466, U.S. Department of Education, National Center for Educational Statistics, U.S. Government Printing Office, Washington, DC.
- [5] Biswas, G., Jeong, H., Kinnebrew, J., Sulcer, B., and Roscoe, R. 2010. Measuring self-regulate learning skills through social interactions in a teachable agent environment. *Research and Practice in Tech-Enhanced Learn*, 5, (Jul 2010), 123-152.
- [6] Clarke-Midura, J., Code, J., Dede, C., Mayrath, M., and Zap, N. 2011. Thinking outside the bubble: Virtual performance assessments for measuring complex learning. In *Technology-based Assessments for 21st century skills: Theoretical and Practical Implications from Modern Research*, M. C. Mayrath, J. Clarke-Midura, & D. Robinson Eds., Information Age, Charlotte, NC, 125-147.
- [7] Craik, F. I. M., and Lockhart, R. S. 1972. Levels of processing: A framework for memory research. *J. of Verb. Learn. and Verb. Beh.* 11,6, (Dec 1972) 671-684.
- [8] Dillon, J. 1988. *Questioning and teaching: A manual of practice*. New York, NY: Teachers College Press.
- [9] Forsyth, C. M., Luce, C., Zapata-Rivera, D., Jackson, G. T., Evanini, K., and So, Y. 2019. *Evaluating English language learners' conversations: Man vs. machine*. *Interna. J. on Comp. Assist. Lang. Learn.* 32, 4 (May 2019), 398-417. DOI= 10.1080/09588221.2018.1517126
- [10] Gholson, B., Witherspoon, A., Morgan, B., Brittingham, J. K., Coles, R., Graesser, A. C., Sullins, J., and Craig, S. D. 2009. Exploring the deep-level reasoning questions effect during vicarious learning among eighth to eleventh graders in the domains of computer literacy and Newtonian physics. *Instruct. Science.* 37, 5 (Sep 2009), 487-493.
- [11] Gil, L., Bråten, I. Vidal-Abarca, E., and Strømsø, H. 2010. Summary versus argument tasks when working with multiple documents: Which is better for whom? *Cont.y Educ. Psych.* 35, 3 (Jul 2010), 157-173. DOI= 35. 157-173. 10.1016/j.cedpsych.2009.11.002.
- [12] Goldman, S. R., Duschl, R. A., Ellenbogen, K., Williams, S., and Tzou, C. T. 2003. Science inquiry in a digital age: Possibilities for making thinking visible. In *Cognition in a digital world*. H. van Oostendorp Ed. Erlbaum, Psychology Press. Mahwah, NJ, 253-284.
- [13] Graesser, A. C. 2016. Conversations with AutoTutor help students learn. *Inter. J. of Artif. Intell. in Educ.* 26,1 (Mar 2016), 124-132.
- [14] Graesser, A. C., Ozuru, Y., and Sullins, J. 2009. What is a good question? *In Threads of coherence in research on the development of reading ability*. M. G. McKeown, & L. Kucan, Eds. Guilford, New York, NY, 112-141.
- [15] Graesser, A. C., and Person, N. K. 1994. Question asking during tutoring. *Amer. Educat. Res. J.* 31,1 (Mar 1994), 104-137.
- [16] Ireland, M. E., Slatcher, R. B., Eastwick, P. W., Scissors, L. E., Finkel, E. J., and Pennebaker, J. W. 2011. Language style matching predicts relationship initiation and stability. *Psych. Sci.* 22,1, (Jan 2011), 39-44.
- [17] Johnson, W. L., Rickel, J. W., and Lester, J. C. 2000. Animated pedagogical agents: Face-to-face interaction in interactive learning environments. *Intern. J. of Artif. Intell. in Educ.* 11,1 (Nov 2000), 47-78.
- [18] Johnson, W. L., and Lester, J. C. 2016. Face-to-face interaction with pedagogical agents, Twenty years later. *Inter. J. of Artif. Intel. in Educ.* 26,1 (Mar 2016), 25-36.
- [19] McNamara, D. S., O'Reilly, T., Best, R., and Ozuru, Y. 2006. Improving adolescent students' reading comprehension with iSTART. *J. of Educ. Comp. Res.*, 34,2 (Mar 2006), 147-171.
- [20] Millis, K., Forsyth, C., Butler, H., Wallace, P., Graesser, A. C., and Halpern, D. 2011. Operation ARIES! A serious game for teaching scientific inquiry. In *Serious Games and Entertainment Applications*, M. Ma, A. Oikonomou, & J. Lakhmi Eds. Springer-Verlag, London, 169-196.
- [21] Moreno, R., Mayer, R. E., Spires, H. A., and Lester, J. 2001. The case for social agency in computer-based teaching: Do students learn more deeply when they interact with animated pedagogical agents? *Cog. and Instruct.* 19, 2 (Jun 2001), 117-213.
- [22] Mosenthal, P. 1996. Understanding the strategies of document literacy and their conditions of use. *J. of Ed. Psych.* 88, 2 (June 1996), 314-332.
- [23] Olney, A., D'Mello, S. K., Person, N., Cade, W., Hays, P., Williams, C., Lehman, B., and Graesser, A. C. 2012. Guru: A computer tutor that models expert human tutors. In *Proceedings of the 11th International Conference on Intelligent Tutoring Systems* (Springer, Berlin, Heidelberg, Jun 14, 2012), ITS2012. Springer-Verlag, Berlin, 256-261
- [24] Pennebaker, J. W., Boyd, R. L., Jordan, K., & Blackburn, K. 2015. *The Development and Psychometric Properties of LIWC2015*. University of Texas at Austin., Austin, TX. DOI: 10.15781/T29G6Z
- [25] Pennebaker, J. W., Chung, C. K., Frazee, J., Lavergne, G. M., and Beaver, D. I. 2014. When small words foretell academic success: The case of college admissions essays. *PLoS One.* 9, 12 (Dec 2014), 110 p. e115844.
- [26] Quellmalz, E. S., Timms, M. J., Buckley, B. C., Davenport, J., Loveland, M., & Silbergitt, M. D. (2011). 21st Century Dynamic Assessment. In M.C. Mayrath, J. Clarke-Midura, & D. Robinson (Eds.), *Technology-based assessments for 21st century skills: Theoretical and practical implications from modern research*. Charlotte, NC: Information Age. 55-90.
- [27] Reder, L. 1987. Strategy selection in question answering. *Cognitive Psychology*, 19, 1 (Jan 1987), 90- 138.
- [28] Rouet, J. 2006. The skills of document use: From text comprehension to web-based learning. Mahwah, NJ: Erlbaum.

- [29] Singer, M. 2003. Processes of question answering. In *Psycholinguistics*, G. Rickheit, T. Hermann, and W. Deutsch Eds. Walter de Gruyter, Berlin, 422-431.
- [30] So, Y., Zapata-Rivera, D., Cho, Y., Luce, C., and Battistini, L. 2015. Using trialogues to measure English language skills. *J. of Educat. Tech. of Edu. Tech. and Society. Special Issue: Tech. Supp. Assess. in Formal and Inform, Learn*, 18, 2 (Apr 2015), 21-32. *Informal Learning* J. García Laborda, D. G Sampson, R. K. Hambleton and E. Guzman (Eds.). 21-32.
- [31] Song, Y., Sparks, J.R., Brantley, W., Oliveri, M., and Zapata-Rivera, D. 2014. Designing Game Activities to Assess Students' Argumentation Skills. *Paper presented at the annual meeting of the American Educational Research Association (AERA)*, Philadelphia, PA.
- [32] Sparks, J.R., Peters, S., Steinberg, J., James, K., Lehman, B.A., & Zapata-Rivera, D. (2019, April). Individual difference measures that predict performance on conversation-based assessments of science inquiry skills. In *Proceedings at the annual meeting of the American Educational Research Association* (Toronto, Canada, April 5-9, 2019).
- [33] Sparks, J.R., and Zapata-Rivera, D. (2019, February). *Designing virtual agents for assessment*. Presentation to Educational Testing Service Constructed Response Forum, Princeton, NJ
- [34] Sparks, J.R., Zapata-Rivera, D., Lehman, B., James, K., and Steinberg, J. 2018. Simulated Dialogues with Virtual Agents: Effects of Agent Features in Conversation-Based Assessments. In *Proceedings of International Conference on Artificial Intelligence in Education*, (Springer, Cham) 469-474. DOI= 10.1007/978-3-319-93846-2_88.
- [35] Ward, W., Cole, R., Bolaños, D., Buchenroth-Martin, C., Svirsky, E., and Weston, T. 2013. My science tutor: A conversational multimedia virtual tutor. *J. of Educ. Psych.* 105, 4 (Nov 2013), 1115-1125.
- [36] Zapata-Rivera, D., Jackson, T., Liu, L., Bertling, M., Vezzu, M., & Katz, I. R. (2014) Science Inquiry Skills using Trialogues. *The Proceedings of 12th International conference on Intelligence Tutoring Systems*. 625-626.

APPENDIX A

Table 2. Final Cluster Center Means and Standard Deviations

Feature	Clus1 (n=52)	Clus2 (n=50)	Clus3 (n=65)	Clus4 (n=66)	F
Word Count	14 (8)	23 (14)	19 (13)	21 (15)	4.6
Analytic Think	5 (6)	39 (28)	14 (22)	43 (30)	34.5
Emotional-Tone	67 (38)	95 (7)	35 (28)	24 (6)	104.2
Word per Sentence	11 (6)	17 (11)	12 (8)	15 (12)	4.3
Dictionary	90 (8)	86 (10)	88 (10)	77 (18)	14.2
function	58 (12)	49 (14)	60 (14)	47 (15)	14.1
personal-pronoun	20 (9)	10 (7)	18 (10)	7 (8)	30.9
“you”	10 (10)	3 (4)	3 (5)	1 (4)	23.0
“She/he”	1 (3)	1 (2)	0 (1)	2 (4)	4.4
article	2 (4)	5 (5)	3 (4)	5 (6)	7.0
prepositions	3 (5)	7 (6)	5 (5)	7 (7)	6.9
auxiliary verb	15 (8)	10 (6)	12 (9)	9 (6)	7.0
negations	1 (3)	1 (3)	11 (19)	2 (5)	12.9
verb	19 (10)	16 (7)	20 (12)	15 (9)	4.1
adjective	12 (9)	10 (9)	6 (7)	8 (8)	5.9
compare	9 (8)	8 (9)	4 (6)	7 (7)	5.3
number	0 (3)	2 (4)	1 (3)	2 (4)	2.9
affect	9 (9)	10 (8)	4 (7)	1 (3)	20.8
Positive emotion	8 (9)	10 (8)	2 (5)	0 (1)	32.6
male	1 (3)	1 (2)	0 (1)	2 (4)	4.2
insight	3 (6)	4 (6)	6 (6)	3 (4)	4.0
drives	10 (9)	9 (7)	4 (6)	2 (4)	19.5
achieve	3 (6)	4 (6)	0 (1)	0 (1)	9.8
reward	3 (6)	5 (7)	1 (2)	0 (1)	17.1
Focus past	1 (3)	1 (3)	3 (5)	1 (3)	4.8
Focus future	3 (5)	2 (3)	1 (2)	1 (3)	3.3

For a complete explanation of each variable, please see the LIWC manual 2015 [30]. The final resulting profiles mentioned above were titled Shallow Performers (Cluster 1), One-Turn Wonders (Cluster 2), Low Performers (Cluster 3) and High Performers (Cluster 4).

APPENDIX B

Table 3. Final Features Correlations to Total Score and Conditions

Feature	Total CBA Score	Question Framing
Word Count	.448**	-.117
Analytic Think	.159*	-.061
EmotionalTone	-.108	.184**
Word per Sentence	.327**	-.067
Dictionary	.033	.278**
function	.022	.160*
personal pronoun	-.122	.144*
“you”	-.225**	.146*
“She/he”	.034	-.251**
article	.148*	-.069
prepositions	.281**	.059
auxiliary verb	.155*	.011
negations	-.177**	.101
verb	.167*	-.010
adjective	.094	-.174**
compare	.097	-.278**
number	.085	-.187**
affect	-.275**	.249**
Positive emotion	-.263**	.217**
male	.040	-.253**
insight	.071	-.132*
drives	-.076	.142*
achieve	-.131*	-.086
reward	-.170**	.025
Focus past	.132*	.042
Focus future	.017	.177**

**p>.01, *p>.05

Does chronology matter? Sequential vs contextual approaches to knowledge tracing

Yueqi Wang
University of California, Berkeley
yueqi@berkeley.edu

Zachary Pardos
University of California, Berkeley
pardos@berkeley.edu

ABSTRACT

Deep learning architectures such as RNN and pure-attention based models have shown state-of-the-art performance in modeling student performance, yet the sources of the predictive power of such models remain an open question. In this paper, we investigate the predictive power of aspects of LSTM and pure attention-based architectures that model sequentiality. We design a knowledge tracing model based on a general transformer encoder architecture to explore the predictive power of sequentiality for attention-based models. For the LSTM-based Deep Knowledge Tracing model, we manipulate the state transition coefficient matrix to turn sequential modeling on and off. All models are evaluated on four public tutoring datasets from ASSISTments and Cognitive Tutor. Experimental results show that DKT and pure-attention based models are overall insensitive towards removing major sequential signals by disabling their sequential modeling parts but with the attention-based model about four times more sensitive. Lastly, we shed light on benefits and challenges of sequential modeling in student performance prediction.

Keywords

Knowledge tracing, DKT, BKT, Cognitive Tutor, ASSISTments, KDD Cup, Contextual embedding, Self-attention

1. INTRODUCTION

Bayesian and Deep Knowledge Tracing models use machine learning architectures with explicit representation of time slices that capture sequentiality. A new generation of models have emerged based on self-attention that have outperformed both BKT and DKT on the knowledge tracing task [9, 11, 19, 17, 7]. Interestingly, the transformer-based architecture that these models rely on has relatively weak modeling of the concept of a time-slice and subsequently of sequence. This prompts us to ask the question of what role sequentiality plays in the predictive performance of the latest generations of neural knowledge tracing models. Some

of the better performing self-attention models adopt a combination of transformer and LSTM architectures, raising the question of if the LSTM is superior for capturing sequence-based signals. To investigate the role of sequentiality in the predictive power of the RNN-based DKT and transformer-based self-attention, or "contextual" models, we design experiments that compare two variants of each model; the original, a version in which time-components of the architecture are systematically disabled. Using four public datasets, we report each model's sensitivity to removing signals of sequence. Our major contributions in this work are:

1. A simple, yet theoretically effective way of disabling attention-based models' sequential modeling components.
2. A novel way of insulating DKT's sequential modeling components to study its sequential modeling ability.
3. We find that DKT and BertKT are exceptionally robust to losing sequential modeling components in terms of performance.

2. RELATED WORK

For studying sequence effects for student performance prediction, Ding and Larson [4] showed that relative ordering of skills have negative effects on the performance of DKT. Kim et al. [10] studied DKT's behavior by conducting a series of perturbation experiments on Monotonicity, Robustness and Convergence etc. Work prior to DKT has also studied sequencing effects, Pardos and Heffernan [13] used a modification of the BKT model to capture learning rates of ordered item pairs, finding that while statistically separable learning rates could be fit, this extra modeling of order did not significantly improve performance prediction accuracy.

Pure-attention based models such as Transformer [18] and Bidirectional Encoder Representations from Transformers (BERT) [3] were first introduced in the NLP field and achieved state of the art performance on major NLP tasks. These models, compared to LSTM predecessors, have the added advantage of enabling parallel training for computational efficiency [18] [2] and being better able to capture long term dependencies. Adapting these strengths to the knowledge tracing task, Self Attentive Knowledge Tracing (SAKT) [12] introduces the earlier attention-based models to solve student performance prediction. SAINT+ [14], which leverages various input features embeddings to assist tutoring on massive online learning platforms, and AKT [7] further improves

Y. Wang and Z. Pardos. Does chronology matter? Sequential vs contextual approaches to knowledge tracing. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 601–605, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852966>

the attention-based architecture by introducing monotonic attention and context-aware distance measure.

Hybrid sequential-contextual models have shown success. *Last Query transformer RNN model* [9] LSTM-SAKT [11] and MUSE [19] utilize the combination of transformer encoder and LSTM/GRU and achieved top-5 performance in a Kaggle AIED Prediction Challenge 2020¹.

3. METHODOLOGY

3.1 Problem Restatement

Our study on sequential effects is based on the task of student performance prediction. We define this task as follows.

Definition 1. Given a student s 's past interactive sequence from time step 1 to $t - 1$ that contains problem marked by required skill ids $X_{t-1}^{(s)}$ and their correctness of responses $Y_{t-1}^{(s)}$, performance prediction of problem x of student s at time step t is defined as:

$$\hat{y}_t^{(s)} = f(X_{t-1}^{(s)}, Y_{t-1}^{(s)}, x_t^{(s)})$$

where $X_{t-1}^{(s)} = \{x_i^{(s)} \in \mathbb{Z}^+ | 1 \leq i \leq t - 1, i \in \mathbb{Z}^+\}$ and $Y_{t-1}^{(s)} = \{y_i^{(s)} \in \mathbb{Z}^+ | 1 \leq i \leq t - 1, i \in \mathbb{Z}^+\}$

3.2 The BertKT Architecture

To study the positional effect of transformer family of models, we design a simple, general architecture named **Bidirectional encoder representations from transformers for Knowledge Tracing**, or **BertKT**. We design this model instead of utilizing previously proposed models for several reasons:

1. It does not contain redundant structures other than transformer encoder layers and an output layer for our downstream task. This helps us isolate the impact of sequential effects for attention-based models.
2. It adopts the most widely used sine-cosine positional encoding with easy enabling/disabling, which facilitates generalization of our results.
3. This architecture is flexible due to its simplicity to accommodate additional settings for future research.

In this work, we conduct student performance prediction in an auto-regressive fashion supported by an upper-triangular attention masks. We do not utilize the bi-directionality here, but such functionality could be enabled for future use such as for bi-directional pre-training tasks. Thus, we keep here the naming convention "bi-directional" as is.

3.2.1 Knowledge and Positional Encoding

In transformer models, scaled-dot-product self-attention which runs in $O(n^2)$ [18]. Thus we adopt word embedding [1] in stead of one-hot encoding in the field of NLP to obtain a

¹<https://www.kaggle.com/c/riiid-test-answer-prediction>

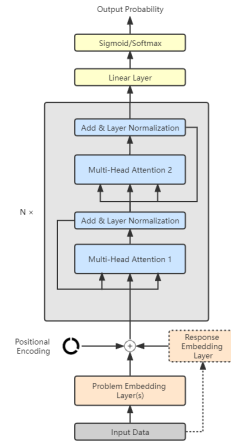


Figure 1: BertKT Architecture Visualization. The shaded part is the self-attention based encoding method.

learnable knowledge encoding with higher computational efficiency. In this work, we only use skill and responses as input features. Embeddings of such features are element-wise-summed together to form a unified representation named combined embedding.

The transformer model is permutation-invariant [18, 5] without positional encoding, making it suffer from not being able to capture original sequence order. A wide varieties of positional encodings are introduced for better performance or interpretability[8, 7]. To study the effect of positionality in this type of model, we completely remove the sine-cosine positional encoding of our model introduced in section 3.2 and compare the model's performance with the baseline model (positional encoding on).

3.2.2 BertKT encoder

In section 3.2.1, we have introduced our element-wise-summed embedding. On top of that, we add a canonical transformer encoder to form the integrated BertKT Architecture. For computation flow, the skill embedding layer and the response embedding layer first take in skill ids and responses to compute their raw embeddings. These two embeddings together with positional encoding are summed element-wise to form a combined embedding. Secondly, the combined embedding goes through BertKT encoder to learn rich contextualized representations. Such contextualized embedding is further processed by linear output layers followed by sigmoid/softmax activation to output response predictions. The entire BertKT architecture can be viewed in Figure 1.

3.3 Manipulations of Sequential Factors

We manipulate the model structures to control the intensity of sequential signals learned in the hidden space. In terms of naming, we add the prefix *static-* to the resulting models with sequential signals modeling disabled.

3.3.1 Static BertKT

In this section, we describe a modification to BertKT's positional encoding to explore sequential influences of the pro-

duced hidden space on the models’ performance. Specifically, we remove completely the sine-cosine positional encoding of BertKT and name it static-BertKT. For any position i in a sequence ranging from $1, \dots, n$ attending to other items, the model can not distinguish the order of those items even if they are manually arranged in the correct order. This is because all other items are treated the same when computing attention regardless of their position. In such a sense, disabling the positional encoding is equivalent to disabling the exclusive indication for the model to capture explicit positionality. We compare the performance of BertKT and static BertKT in section 6.1.

3.3.2 Static DKT

In this section, we describe our modification to the sequential LSTM structures in DKT to observe performance changes due to effectively disabling weights associated with sequentiality modeling.

Static DKT Experiment

For DKT and all its variants we utilize in this work we employ an LSTM as the core computational structure. The formula for a canonical LSTM structure is as follows.

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \quad (1)$$

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \quad (2)$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \quad (3)$$

$$\tilde{c}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c) \quad (4)$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \tilde{c}_t \quad (5)$$

$$h_t = o_t \circ \tanh(c_t) \quad (6)$$

where

$x_t \in \mathbb{R}^d$: input vector of time step t

$f_t \in (0, 1)^h$: forget gate

$i_t \in (0, 1)^h$: input gate

$o_t \in (0, 1)^h$: output gate

$h_t \in (-1, 1)^h$: hidden state vector of time step t

$\tilde{c}_t \in (-1, 1)^h$: cell input

$c_t \in \mathbb{R}^h$: cell state

$W \in \mathbb{R}^{h \times d}$, $U \in \mathbb{R}^{h \times h}$ and $b \in \mathbb{R}^h$: learnable weight and bias; d and h are the dimension of input features and that of hidden states.

Concretely, the hidden state of time step $t + 1$ is dependent on that of t , the current input item at $t + 1$, but not on any future time step ranging from $[t + 2, \dots, T]$, where T is the ending time step. Here we would like to disable the sequential, hidden-state transitioning dynamics to compare results with the original DKT model. Concretely, we set each of $U_f, U_i, U_o, U_c \in \mathbb{R}^{h \times h}$ to an identity matrix $I \in \mathbb{R}^{h \times h}$. Obviously for any hidden state h , an identity matrix multiplication equal to no transformation exerted at all. Also the h could be combined with the bias term b . However, we still keep matrix U here for 2 reasons:

1. For the completeness of math formulation as this is a special case of the dynamic LSTM with learnable U .
2. In terms of implementation, U could be changed easily from a placeholder to other values for future research experiments.

Table 1: Model names and the corresponding explanations

Model/Setting	-
BertKT	baseline BertKT
static-BertKT	BertKT with positional encoding removed
DKT	baseline DKT
static-DKT	DKT with static hidden state update matrix

In the end, we have the following *static-LSTM* as the new component of the corresponding DKT model we refer to as static-DKT.

$$f_t = \sigma(W_f x_t + I h_{t-1} + b_f) \quad (7)$$

$$i_t = \sigma(W_i x_t + I h_{t-1} + b_i) \quad (8)$$

$$o_t = \sigma(W_o x_t + I h_{t-1} + b_o) \quad (9)$$

$$\tilde{c}_t = \tanh(W_c x_t + I h_{t-1} + b_c) \quad (10)$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \tilde{c}_t \quad (11)$$

$$h_t = o_t \circ \tanh(c_t) \quad (12)$$

where each gate update function remains the same but the explicit hidden state update is disabled.

Note that we do not hamper the sequential update format of cell state c and hidden state h , as they are part of the nature of a classic LSTM model. The hidden state is still sequentially updated but only implicitly through W_f , W_i , W_c and W_o . In summary, this experiment is to explore to what extent that sequential modeling factor of DKT contributes to its total predictive power in terms of evaluations metrics we introduce in section 5.2.

4. COMPARISON MODELS

In this section, we provide a reference table 1 to summarize the name of each model and their corresponding settings.

5. EXPERIMENTAL SETUP

5.1 Datasets

We evaluate the performance of BertKT and other comparison models based on four benchmark datasets: ASSISTments 2009-2010² [6], ASSISTments 2012-2013³, Cognitive Tutor Bridge to Algebra 2006-2007⁴ and Cognitive Tutor Bridge to Algebra 2008-2009⁵ [15]. Among these benchmarks, ASSISTments datasets are collected from ASSISTments online tutoring system primarily for secondary school mathematics. Cognitive Tutor (now MATHia) is an intelligent tutoring system and the datasets we used are from its Algebra curricula. The two Cognitive Tutor datasets we use were used as the official development/competition dataset for the KDD Cup 2010 Challenge [16].

For all the above datasets, we follow a series of conventional pre-processing procedures such as removing all problems not

²<https://drive.google.com/file/d/1NNXHFRxcArrU0ZJSb9BIL56vmUt5Fh1E/view?usp=sharing>

³<https://drive.google.com/file/d/1cU6Ft4R3hLqA7G1rIGArVfelSZvc6RrXy/view?usp=sharing>

⁴<http://pslcdatashop.web.cmu.edu/KDDCup/downloads.jsp>

⁵<http://pslcdatashop.web.cmu.edu/KDDCup/downloads.jsp>

Table 2: Comparison between default BertKT and static-BertKT. Best metrics are in bold.

Model	BertKT				static-BertKT			
	BCE	AUC	ACC	RMSE	BCE	AUC	ACC	RMSE
ASSISTments 2009	0.4948	0.8149	0.7562	0.4055	0.4932	0.8166	0.7560	0.4051
ASSISTments 2012	0.5457	0.7209	0.7309	0.4274	0.5453	0.7203	0.7322	0.4272
Cog Tutor 2006	0.3724	0.7872	0.8462	0.3380	0.3741	0.7842	0.8456	0.3387
Cog Tutor 2008	0.2698	0.9055	0.8881	0.2869	0.2765	0.9010	0.8864	0.2895

Table 3: Comparison between default DKT and static-DKT. Best metrics are in bold.

Model	DKT				static-DKT			
	BCE	AUC	ACC	RMSE	BCE	AUC	ACC	RMSE
ASSISTments 2009	0.4981	0.8109	0.7542	0.4070	0.4975	0.8108	0.7548	0.4066
ASSISTments 2012	0.5402	0.7253	0.7350	0.4249	0.5400	0.7256	0.7354	0.4247
Cog Tutor 2006	0.3721	0.7870	0.8461	0.3378	0.3712	0.7881	0.8471	0.3372
Cog Tutor 2008	0.2679	0.9063	0.8881	0.2864	0.2675	0.9064	0.8883	0.2862

associated with a concept/skill and using correct on first attempt responses only.

5.2 Models and Evaluation Metrics

We compare BertKT and DKT with their sequentiality-weakened counterparts. Differences and features of these model have been discussed in section 4; We evaluate each model across four metrics: BCE (Binary cross entropy loss), AUC (Area under the receiver operating characteristic curve), ACC (Accuracy as the proportion of correct classification with threshold 0.5) and RMSE (Root-mean-square error).

5.3 Training and Testing

We take out 20% of the entire data as the development set to conduct hyper-parameter tuning. For training and testing, We perform student-level k -fold cross validation (with $k = 5$) on each model, meaning each student will appear exclusively in training, validation or test set. In each phase of the cross validation, 60% of the dataset is used as the training set, 20% as the validation set to perform early stopping, and the rest 20% as the test set.

We fix a sequence length of 100 for computational efficiency with longer sequences split to subsequences of length 100 and trailing subsequence padded. We use Adam optimizer to update parameters for all models trained on two NVIDIA GeForce RTX 2080 Ti (11 GB) or TITAN Xp GPU (12 GB). All models are implemented using PyTorch with fixed random seeds for reproducibility.

6. EXPERIMENTAL RESULTS

In this section, we compare evaluations results of BertKT and DKT with their variants introduced in section 3. We report evaluation metrics per each model per metrics defined in section 5.2 as well as the relative percentage increase of these metrics between comparison models. Models with the best metrics are in bold. Specifically, we report the percentage increase by the "performance increase/decrease", meaning that increase in {BCE, RMSE} is marked by "-" and that increase in {AUC, ACC} is marked by "+". Decrease means the opposite way.

6.1 BertKT Model Results

Table 2 shows the evaluations metrics for BertKT and static-BertKT. The best performances by metric are in bold. We

have explained the meanings of these variant models in table 1. Overall, the baseline BertKT performs better, with 10 bolded cells, than its sequentially disrupted variants. However, the percentage performance differences averaged over the four dataset benchmarks of static-BertKT from BertKT are -0.64%, -0.19%, -0.02%, -0.24% for BCE, AUC, ACC and RMSE, respectively ('+' indicates a performance improvement for the metric, while '-' represents a performance decrease). This suggests that BertKT is insensitive towards sequential permutation even without positional encoding.

6.2 DKT Model Results

In this section, we compare in table 3 DKT with its sequentially weakened counterpart, namely, DKT and static-DKT. Naming conventions have been introduced in table 1 in section 4.

Surprisingly, the static-DKT model was the best performing model on all datasets for all but one metric of one dataset (ASSISTments 2009-AUC). The average percentage performance differences averaged over the four dataset benchmarks of static-DKT from DKT are +0.13%, +0.04%, +0.07%, +0.09% for BCE, AUC, ACC and RMSE, respectively ('+' indicates a performance improvement for the metric, while '-' represents a performance decrease). Even after disabling weights associated with time-slice transitions, the model performs closely to the baseline model and even with a slight increase. This suggests that the embedding weights for input X W_f , W_i , W_o and W_c are still sufficient to capture signals that support the model's original performance.

Across the four metrics and four datasets, DKT improves 0.08% due to a static hidden state transition, while the performance of BertKT decreases by 0.27% due to no positional encoding (static). The average percentage performance loss of BertKT due to being static is -4.3340x larger than that of DKT. This suggests BertKT's positional encoding is more sensitive in modeling the sequence order than DKT in terms of performance.

7. CONCLUSIONS AND DISCUSSION

While our motivating observation was that transformer-based models have weaker explicit modeling of time than LSTM-based models, we find that BertKT is more dependent on its positional encoding than DKT is on its time-slice transition weights. Such sensitivity may be an indication that there is room for improvement for positional encodings to better leverage input sequence order. This improvement is perhaps seen in the performance of AKT [7], which outperforms DKT and employs additional modeling of sequentiality in the form of imposed monotonicity of attention weights with respect to time.

Our results raise the question of why DKT is not negatively affected by disabling its transition weights. Potentially, the temporal signal is being pushed down into the embedding W . Is student growth better captured in the interactions between input embeddings than it is by a generalized recurrent hidden state transition? This may speak to the powerful but simple assumption of the original knowledge tracing model, that learning is a function of *opportunity count*, growth rate, and prior.

8. REFERENCES

- [1] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin. A neural probabilistic language model. *The journal of machine learning research*, 3:1137–1155, 2003.
- [2] C. Chen and Z. Pardos. Applying recent innovations from nlp to mooc student course trajectory modeling. In *Proceedings of the 13th International Conference on Educational Data Mining*, 2020.
- [3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [4] X. Ding and E. C. Larson. Why deep knowledge tracing has less depth than anticipated. In *Proceedings of the 12th International Conference on Educational Data Mining*, 2019.
- [5] P. Dufter, M. Schmitt, and H. Schütze. Position information in transformers: An overview. *arXiv preprint arXiv:2102.11090*, 2021.
- [6] M. Feng, N. Heffernan, and K. Koedinger. Addressing the assessment challenge with an online system that tutors as it assesses. *User Modeling and User-Adapted Interaction*, 19(3):243–266, 2009.
- [7] A. Ghosh, N. Heffernan, and A. S. Lan. Context-aware attentive knowledge tracing. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2330–2339, 2020.
- [8] Z. Huang, D. Liang, P. Xu, and B. Xiang. Improve transformer models with better relative position embeddings. In *Findings of Empirical Methods in Natural Language Processing*, 2020.
- [9] S. Jeon. Last query transformer rnn for knowledge tracing. In *Workshop on AI Education@35th AAAI Conference on Artificial Intelligence*, 2021.
- [10] M. Kim, Y. Shim, S. Lee, H. Loh, and J. Park. Behavioral testing of deep neural network knowledge tracing models. In *Proceedings of the 14th International Conference on Educational Data Mining*, 2021.
- [11] T. Oya and S. Morishima. Lstm-sakt: Lstm-encoded sakt-like transformer for knowledge tracing. In *Workshop on AI Education@35th AAAI Conference on Artificial Intelligence*, 2021.
- [12] S. Pandey and G. Karypis. A self-attentive model for knowledge tracing. In *Proceedings of the 12th International Conference on Educational Data Mining*, 2019.
- [13] Z. A. Pardos and N. T. Heffernan. Determining the significance of item order in randomized problem sets. In T. Barnes, M. C. Desmarais, C. Romero, and S. Ventura, editors, *Educational Data Mining - EDM 2009, Cordoba, Spain, July 1-3, 2009. Proceedings of the 2nd International Conference on Educational Data Mining*, pages 111–120. www.educationaldatamining.org, 2009.
- [14] D. Shin, Y. Shim, H. Yu, S. Lee, B. Kim, and Y. Choi. Saint+: Integrating temporal features for ednet correctness prediction. In *Proceedings of the 11th International Learning Analysis and Knowledge Conference*, 2021.
- [15] J. Stamper, A. Niculescu-Mizil, S. Ritter, G. Gordon, and K. Koedinger. Bridge to algebra 2008-2009. challenge data set. In *KDD Cup 2010 Educational Data Mining Challenge*, 2010.
- [16] J. Stamper and Z. A. Pardos. The 2010 kdd cup competition dataset: Engaging the machine learning community in predictive learning analytics. *Journal of Learning Analytics*, 3(2):312–316, 2016.
- [17] D. K. L. Tran. Riiid! answer correctness prediction kaggle challenge: 4th place solution summary. In *Workshop on AI Education@35th AAAI Conference on Artificial Intelligence*, 2021.
- [18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.
- [19] C. Zhang, Y. Jiang, W. Zhang, and C. Gu. Muse: Multi-scale temporal features evolution for knowledge tracing. In *Workshop on AI Education@35th AAAI Conference on Artificial Intelligence*, 2021.

Algorithmic unfairness mitigation in student models: When fairer methods lead to unintended results

Frank Stinar
University of Illinois Urbana–Champaign
fstinar2@illinois.edu

Nigel Bosch
University of Illinois Urbana–Champaign
pnb@illinois.edu

ABSTRACT

Systematically unfair education systems lead to different levels of learning for students from different demographic groups, which, in the context of AI-driven education, has inspired work on mitigating unfairness in machine learning methods. However, unfairness mitigation methods may lead to unintended consequences for classrooms and students. We examined preprocessing and postprocessing unfairness mitigation algorithms in the context of a large dataset, the State of Texas Assessments of Academic Readiness (STAAR) outcome data, to investigate these issues. We evaluated each unfairness mitigation algorithm across multiple machine learning models using different definitions of fairness. We then evaluated how unfairness mitigation impacts classifications of students across different combinations of machine learning models, unfairness mitigation methods, and definitions of fairness. On average, unfairness mitigation methods led to a 22% improvement in fairness. When examining the impacts of unfairness mitigation methods on predictions, we found that these methods led to models that can and did overgeneralize groups. Consequently, predictions made by such models may not reach the intended audiences. We discuss the implications for AI-driven interventions and student support.

Keywords

Machine learning, Unfairness mitigation, Fairness, Data science applications in education

1. INTRODUCTION

Student assessment has been plagued with biases against students from different demographic groups [2]. Various fairness metrics have been designed to evaluate the severity of biases [27], including biases in automatic assessments powered by machine learning [13]. Researchers have explored methods for mitigating such biases [3, 16, 21], including for assessment-related tasks such as student grade prediction [18]. However, the implications of applying unfairness mit-

igation methods to educational datasets are currently not well understood. In this paper, we explore unfairness mitigation on the State of Texas Assessments of Academic Readiness (STAAR) dataset [25], including examples of what impacts might be felt by students when models are optimized for different definitions of fairness.

The influx of data stemming from education contexts has allowed computational methods to be used for understanding education [4]. Alongside the creation of computer-based assessment methods, researchers have also analyzed current assessment data with machine learning and other computational methods [17]. These forms of analysis allowed researchers to uncover new biases towards different demographic groups in education and assessment methods [3]. Moreover, newer methods do not necessarily fix the biases present in traditional assessments [21]. However, the newer methods are needed when dealing with complex (e.g., nonlinear, multimodal) educational datasets—as are new methods to address biases present in computational analyses [7, 8].

Biases in the field of education have differing definitions, differing impacts, and can come from different places within educational and social systems [3]. In the study presented in this paper we are unable to disentangle the systemic biases present in assessment systems, as is often the case with educational data that concerns specific contexts, learning environments, or topics. Instead, we focused on biases in statistical measurements and biases present in the machine learning models. In particular, we investigate how unfairness mitigation methods and machine learning models impact student representations in data (and predictions), and if the biases found in these methods are analogous to biases in assessments—which may come from the assessments themselves or from systemic biases. In this paper, we explore these issues by addressing two research questions:

Research Question 1 (RQ1): Do demographic differences in standardized assessment scores correspond to biases in machine learning models? Standardized tests often contain biases [21]. Machine learning models have been shown to exacerbate biases that are present in training data [5]. RQ1 therefore examines the link between test score differences (including biases) and the biases that are found in machine learning models trained on closely related data from the same students. By comparing the biases that machine learning models elicit and the biases present in standardized test data, we are able to disentangle some of the sources of bias

F. Stinar and N. Bosch. Algorithmic unfairness mitigation in student models: When fairer methods lead to unintended results. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 606–611, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853135>

that occur in data-driven assessment, and thereby inform future work on mitigating such biases. In alignment with previous work [3], we expect that biases present in standardized assessments and machine learning models will not perfectly align. If the biases do not align, differences between them indicate that new biases may be introduced—or, ideally, that there are opportunities for reducing such biases.

Research Question 2 (RQ2): What are the implications of machine learning unfairness mitigation methods when applied to student test score prediction? Researching this provides insight into the real-world implications for applying fairness metric optimization to human-based datasets. These implications can provide insight into the types of interventions that could take place in learning environments, because applying machine learning (or even expert-created) models to the real world without substantively examining the possible impacts can lead to dire consequences [1, 5].

By answering these two RQs, we tie test score biases and machine learning biases together through unfairness mitigation strategies and examine the implications of applying these strategies in education. By comparing the outputs of unfairness-mitigated machine learning models with the actual biases present in standardized tests we found a relationship between the two. Then, we showed how learning environments and students are impacted from applying fairness processes. Next, we address related research to position our work in the space of educational data mining and AI fairness research.

2. RELATED WORK

We focus on work related to fair AI methods and fair AI used within educational data mining. For a more general overview, see Fischer et al. [12]; we discuss fair AI, specifically, in the remainder of this section.

We first discuss fairness research being done in AI, regardless of domain. Many different research approaches have differing definitions of fairness and what fair AI looks like. For example, Hu et al. increased fairness in part through comparing differing groups positive predictive rates [16]; others have examined several different statistical definitions of fairness [6], which can even be contradictory with each other [11]. Work has also been done to *explain* many of the definitions of fairness that are used in fair AI research [26], independent of domain.

There is an increasing body of work in the field of educational data mining that uses fair AI, including the impact caused by algorithmic bias in education systems [3, 20]. This research being done in educational data mining with fair AI methods shows the growing use of machine learning in education research, complementing existing work in related topics like fair assessment (e.g., [21, 2]). These publications represent how biases are seen within different data mining avenues.

Using similar connections made in previous work between fair AI methods and educational data mining, we harness machine learning models and unfairness mitigation methods to examine performance differences between demographic groups in standardized tests.

Table 1: Breakdown of Demographic Groups

Demographic Identifier	# Occurrences	% Sample
Female	672,545	17.7%
Male	670,664	17.6%
Economic disadvantage	648,716	17.1%
Hispanic	554,697	14.6%
White	440,972	11.6%
Special education	315,072	8.3%
African American	240,901	6.3%
Asian	125,308	3.3%
Two or more races	80,817	2.1%
Pacific Islander	1,273	0.03%
American Indian	1,050	0.02%

3. METHODS

We used the assessment database from the STAAR Texas Education Agency dataset. These data were collected from the Teaching Trust (a now defunct leadership development group with the goal of eliminating opportunity gaps for students) between 2012 and 2019 [25]. This dataset has information from over 5 million students, which is approximately 10% of the public school students in the United States. The data include at most one demographic identifier per student—e.g., a student’s race or gender might be included, but not both. Table 1 contains the breakdown of demographic-related information in the dataset.

3.1 Machine Learning Models

The three machine learning models we used were the logistic regression, random forest, and extremely randomized trees models. We used 5-fold cross validation and tuned hyperparameters via grid search. We trained all models using `scikit-learn` [22]. The logistic regression used the default hyperparameters from `scikit-learn` (i.e., a small L_2 regularization penalty). The random forest and extremely randomized trees models underwent hyperparameter tuning for the maximum depth of trees and the proportion of features samples for each tree.

3.2 Model Evaluation

For each of the machine learning models we used four unfairness mitigation methods (described in more detail below): disparate impact preprocessing, reweighing, equalized odds postprocessing, and calibrated equalized odds postprocessing. We evaluated each of these model/method combinations in terms of accuracy measured via area under the receiver operating characteristic curve (AUC) and four unfairness metrics: statistical parity difference, disparate impact ratio, average odds difference, and equal opportunity difference (described in the Appendix).

3.3 Unfairness Mitigation Algorithms

We implemented unfairness mitigation algorithms with the AIF360 Python library [7].

3.3.1 Disparate Impact Preprocessing

Disparate impact preprocessing compares the label (passing the STAAR test) base rate across groups. The algorithm

takes this rate and edits features of the original data so that it is impossible to tell which group an individual belongs to.

3.3.2 Reweighting

The reweighting algorithm adds weight to each example during model training based upon the proportion of students in different demographic groups and outcome (e.g., positive vs. negative class) groups. The equation for the weight is given in Equation 1.

$$w_{positive/group1} = \frac{(N_{group1})(N_{positive})}{(N_{all})(N_{positive/group1})} \quad (1)$$

3.3.3 Equalized Odds Postprocessing

Equalized odds postprocessing works to optimize the equalized odds fairness metric by changing predicted labels as needed to satisfy the metric. Specifically, the algorithm solves a linear program for probabilities. From these probabilities, classification labels are given [15]. An equalized odds predictor is made for this program from predicting on equalized odds incentive measurements for all classes.

3.3.4 Calibrated Equalized Odds Postprocessing

Calibrated equalized odds postprocessing follows a similar process to equalized odds postprocessing; however, it optimizes for equalized odds over a calibrated model output. Calibrated output is found when probability predictions align with the actual probability of observing the predicted outcomes [23].

4. RESULTS

We describe our results with respect to the two research questions outlined in the Introduction.

4.1 RQ1: Machine Learning Bias compared to Assessment Bias

RQ1 asks if biases found in machine learning models are analogous to the biases present in the STAAR assessment. Table 2 contains the values of accuracy and fairness metrics found after applying unfairness mitigation algorithms.

On average we observed a 22% improvement in fairness metric evaluation. However, different unfairness mitigation methods led to different trends in results. We found the disparate impact preprocessing method had the smallest impact, on average, across all fairness metrics. However, disparate impact preprocessing yielded the highest accuracy for each model, perhaps because it impacted the models the least. For both the random forest and extremely randomized trees models, reweighting unfairness mitigation led to the fairest predictions across all fairness metrics. We found that for some unfairness mitigation methods, especially equalized odds and calibrated equalized odds, the predictions were so influenced that the accuracy was no better than chance level (i.e., $AUC \leq .500$).

4.2 RQ2: Unfairness Mitigation Implications

We answered RQ2 by analyzing each unfairness mitigation algorithm's impact on the STAAR data or models being used. The disparate impact preprocessing algorithm removed distinctions between groups in the dataset itself. The

disparate impact removal process preserves within-group ranking of singular data points; however, group membership of singular data points are changed so it is not possible to discern what groups individuals belong to. The reweighting algorithm adds weights to different data points based on frequency of group membership to remove bias. Equalized odds postprocessing and calibrated equalized odds postprocessing do not edit the original dataset. Instead, they change output labels of singular data points with the objective of maximizing the equalized odds of the classifications. Each algorithm thus impacts either the student data itself or the process used to classify students as passing or failing the assessment, the implications of which need to be understood to determine whether such methods are procedurally fair—that is, whether the process by which decisions are made is fair, not just the fairness of the decisions themselves [9, 6, 14].

We found that using these unfairness mitigation algorithms resulted in unintended consequences for how students were computationally represented. For example, after disparate impact preprocessing, students who originally belonged to one group were now considered part of a different group for training. In the case of STAAR data, if the data were bimodal, with each group having their own pattern of demographic identifiers, students would be represented in data with different demographic identifiers than they actually have.

The impacts of each unfairness mitigation method are seen in the manipulation of the STAAR data. For example, some individuals in the “Female” demographic group were now in the “Asian” demographic group after disparate impact preprocessing; the algorithm preserved the within-group ranking of individuals, but changed group membership in unintended ways. This may have created a more fair model in terms of predictions, but while ignoring or confusing the systemic education problem present in the data.

In contrast, the two equalized odds based algorithms effectively changed whether or not students in the training data passed or failed the assessment. This may be less drastic than shifting students' demographic identifiers; however, decisions made upon this unfairness mitigation are imperfect if they obfuscate the problem of unequal learning.

Finally, the reweighting algorithm introduces weights to students for fairer classification. This method does not change the students' features, and thus might be considered a more faithful representation of students. However, weighting students may cause unintended effects as demographic group and assessment score combinations become more impactful to classification of others. This is especially true for students from smaller groups, who may find their characteristics or behaviors become far more important to a model's decisions than is desirable.

5. DISCUSSION

Our first research question predicted that demographic differences in test scores do correspond to biases seen in machine learning models trained on those data. Our results show that there were similarities between the biases present; however, models can further propagate biases present in the

Table 2: Fairness metric calculations. The *No Model* row represents the metrics calculated on the original dataset (for metrics that could be calculated from outcome labels in the dataset) for comparison to the rest of the algorithms.

Model	Unfairness Mitigation	AUC	Stat Parity Diff	Disp Imp Ratio	Avg Odds Diff	EqOpp Diff
No Model	None	–	-.172	.766	–	–
LogReg	None	.579	.113	1.419	.124	.162
RandFor	None	.623	-.621	0.379	-.639	-.529
Extra-Trees	None	.623	-.692	0.373	-.614	-.517
LogReg	Disparate Impact	.524	.112	1.427	.122	.161
LogReg	Reweighting	.474	.286	2.071	.293	.348
LogReg	Equalized Odds	.384	.306	1.875	.323	.333
LogReg	Calibrated Equalized Odds	.483	.483	2.391	.503	.486
RandFor	Disparate Impact	.512	-.675	0.325	-.692	-.586
RandFor	Reweighting	.522	.114	1.226	.123	.176
RandFor	Equalized Odds	.471	-.760	0.208	-.769	-.687
RandFor	Calibrated Equalized Odds	.472	-.438	0.232	-.425	-.398
Extra-Trees	Disparate Impact	.528	-.686	0.258	-.698	-.606
Extra-Trees	Reweighting	.518	.081	1.176	.089	.145
Extra-Trees	Equalized Odds	.436	-.605	0.291	0.605	-.539
Extra-Trees	Calibrated Equalized Odds	.445	-.665	0.235	-.668	-.597

data when trained on unfairness-mitigated data. This aligns with other research on biases present in education [24].

Different types of biases can be measured in our machine learning models that can not be measured within STAAR. Models might include biases that are separate from biases present in the test scores, indicating that machine learning models may exacerbate biases already present in data or even introduce new biases. Additionally, putting assessment data into a machine learning pipeline with unfairness mitigation can lead to negative implications as theorized by RQ2.

Our expectation for RQ2 was that applying unfairness mitigation strategies would lead to unintended real-world consequences for students. Indeed, unfairness mitigation strategies manipulated data in ways that could be perceived as unfair, especially with respect to procedural fairness. We investigated this by examining how unfairness mitigation impacted the STAAR data and the classifications made. RQ2 results show if one is planning to administer data-driven interventions in education, applying unfairness mitigation will likely overgeneralize groups. Thus, students who require intervention may not receive it and students who do not need intervention may receive one, not because of model inaccuracies necessarily but because of unfairness mitigation strategies. For example, if students with differing amount of background knowledge are misrepresented, learning outcomes could be negatively impacted [19]. This imprecision of groups that comes with unfairness mitigation can lead to unintended consequences when applied to real-world applications. Thus, unfairness mitigation methods must be applied with caution.

5.1 Limitations and Future Work

The study in this paper explored one prediction task, which—though representative of a large proportion of U.S. students—

is not representative of all assessments nor any of the other educational outcomes and constructs of possible interest. Similarly, we examined a few machine learning models with a selection of preprocessing and postprocessing unfairness mitigation algorithms. We focused on methods that are common in (or well suited to) education data contexts, but the space of possible models and unfairness mitigation algorithms is far larger. Thus, future work would benefit from working with other educational datasets, machine learning models, and unfairness mitigation algorithms to further examine how these methods can impact the representations of students in data analysis. Finally, a nearly insurmountable limitation of this work is that we are unable to disentangle the systemic biases that lead to different amounts of learning (e.g., structural racism and classism) versus the assessment biases (e.g., lack of cultural responsiveness) and algorithmic biases that contribute to biased measurement. We are unable to resolve this problem, but we do evaluate the alignment of these biases in this paper, and suggest that future work with quasi-experimental analyses may be one possible route to address this limitation.

5.2 Conclusion

Education is plagued with unfairness for differing demographic groups [10]. Unfairness mitigation methods have potential to reduce unfairness in data-driven assessment and student support, but when applied to educational datasets these methods may lead to unintended negative consequences. We explored machine learning pipelines with unfairness mitigation methods applied, and examined how these methods would affect the representations of individual students. We expect that these findings will guide the selection of unfairness mitigation methods in future work, and hope that with our findings in mind, when decisions are made from models based on educational data, less harm is done to students from a lack of caution when choosing models and algorithm.

6. REFERENCES

- [1] J. Anders, C. Dilnot, L. Macmillan, and G. Wyness. Grade expectations: How well can we predict future grades based on past performance? CEPEO Working Paper Series 20-14, UCL Centre for Education Policy and Equalising Opportunities, 2020.
- [2] K. Arbutnot. *Filling in the blanks: Understanding standardized testing and the Black-White achievement gap*. IAP Information Age Publishing, Charlotte, NC, US, 2011.
- [3] R. S. Baker and A. Hawn. Algorithmic bias in education. *International Journal of Artificial Intelligence in Education*, 2021.
- [4] R. S. Baker and K. Yacef. The state of educational data mining in 2009: A review and future visions. *Journal of Educational Data Mining*, 1(1):3–17, Oct. 2009. The file is in PDF format. If your computer does not recognize it, simply download the file and then open it with your browser.
- [5] M. Barenstein. ProPublica’s COMPAS data revisited, 2019.
- [6] C. Belitz, L. Jiang, and N. Bosch. Automating procedurally fair feature selection in machine learning. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, page 379–389, New York, NY, USA, 2021. Association for Computing Machinery.
- [7] R. K. E. Bellamy, K. Dey, M. Hind, S. C. Hoffman, S. Houde, K. Kannan, P. Lohia, J. Martino, S. Mehta, A. Mojsilović, S. Nagar, K. N. Ramamurthy, J. Richards, D. Saha, P. Sattigeri, M. Singh, K. R. Varshney, and Y. Zhang. AI Fairness 360: An extensible toolkit for detecting and mitigating algorithmic bias. *IBM Journal of Research and Development*, 63(4/5):1–15, 2019.
- [8] S. Bird, M. Dudík, R. Edgar, B. Horn, R. Lutz, V. Milan, M. Sameki, H. Wallach, and K. Walker. Fairlearn: A toolkit for assessing and improving fairness in AI. Technical Report MSR-TR-2020-32, Microsoft, May 2020.
- [9] K. Burke and S. Leben. Procedural fairness: A key ingredient in public satisfaction. *Court Review*, 44(1-2):4–25, 2007.
- [10] H.-Y. S. Cherng, P. F. Halpin, and L. A. Rodriguez. Teaching bias? Relations between teaching quality and classroom demographic composition. *American Journal of Education*, 128(2):000–000, 2022.
- [11] I. Cojuharenco and D. Patient. Workplace fairness versus unfairness: Examining the differential salience of facets of organizational justice. *Journal of Occupational and Organizational Psychology*, 86:371–393, 2013.
- [12] C. Fischer, Z. A. Pardos, R. S. Baker, J. J. Williams, P. Smyth, R. Yu, S. Slater, R. Baker, and M. Warschauer. Mining big data in education: Affordances and challenges. *Review of Research in Education*, 44(1):130–160, 2020.
- [13] J. Gardner, M. O’Leary, and L. Yuan. Artificial intelligence in educational assessment: ‘Breakthrough? Or buncombe and ballyhoo?’. *Journal of Computer Assisted Learning*, 37(5):1207–1216, 2021.
- [14] N. Grgić-Hlača, M. B. Zafar, K. P. Gummadi, and A. Weller. Beyond distributive fairness in algorithmic decision making: Feature selection for procedurally fair learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, pages 51–60, Apr. 2018.
- [15] M. Hardt, E. Price, and N. Srebro. Equality of opportunity in supervised learning. volume abs/1610.02413, 2016.
- [16] Q. Hu and H. Rangwala. Towards fair educational data mining: A case study on detecting at-risk students. In *Proceedings of the 13th International Conference on Educational Data Mining (EDM 2020)*, pages 431–437. International Educational Data Mining Society, 2020.
- [17] S. Hussain, N. Abdulaziz Dahan, F. Ba-Alwib, and R. Najoua. Educational data mining and analysis of students’ academic performance using WEKA. *Indonesian Journal of Electrical Engineering and Computer Science*, 9:447–459, 02 2018.
- [18] W. Jiang and Z. A. Pardos. Towards equity and algorithmic fairness in student grade prediction. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, page 608–617, New York, NY, USA, 2021. Association for Computing Machinery.
- [19] S. Kalyuga. Expertise reversal effect and its implications for learner-tailored instruction. *Educational Psychology Review*, 19(4):509–539, Dec. 2007.
- [20] R. F. Kizilcec and H. Lee. Algorithmic fairness in education. In W. Holmes and K. Porayska-Pomsta, editors, *Ethics in Artificial Intelligence in Education*. Taylor & Francis, in press.
- [21] R. Mendoza-Denton. A social psychological perspective on the achievement gap in standardized test performance between white and minority students: Implications for assessment. *The Journal of Negro Education*, 83(4):465–484, 2014.
- [22] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [23] G. Pleiss, M. Raghavan, F. Wu, J. M. Kleinberg, and K. Q. Weinberger. On fairness and calibration. volume abs/1709.02012, 2017.
- [24] J. G. Starck, T. Riddle, S. Sinclair, and N. Warikoo. Teachers are people too: Examining the racial bias of teachers compared to other american adults. *Educational Researcher*, 49(4):273–284, 2020.
- [25] T. Texas Education Agency. STAAR. <https://rptsvr1.tea.texas.gov/perfreport/account/2019/download/acctref.html>, 2019.
- [26] S. Verma and J. Rubin. Fairness definitions explained. In *Proceedings of the International Workshop on Software Fairness, FairWare ’18*, page 1–7, New York, NY, USA, 2018. Association for Computing Machinery.
- [27] M. B. Zafar, I. Valera, M. G. Rodriguez, and K. P. Gummadi. Fairness constraints: Mechanisms for fair classification. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*,

APPENDIX

A. FAIRNESS METRICS

We measured fairness according to four quantitative definitions, detailed below. The combination of these metrics allowed us to holistically judge the impact of the unfairness mitigation algorithms on the machine learning models.

A.1 Statistical Parity Difference

The statistical parity difference metric compares the difference between the groups' probability of being predicted to pass the assessment. A value of 0 indicates both groups were predicted to pass the assessment with equal probability. A positive value means one group is predicted to pass the assessment more, a negative value means the other group is predicted to pass the assessment more.

$$SP = P(\hat{Y} = 1|D = group2) - P(\hat{Y} = 1|D = group1) \quad (2)$$

A.2 Disparate Impact Ratio

Disparate impact ratio is the ratio of the differing groups being predicted to pass the assessment. A value of 1 indicates that both groups are predicted to pass the assessment with equal probability. A value greater than 1 indicates that one group is predicted to pass the assessment more than the other group, while a value less than 1 indicates the opposite.

$$DI = \frac{P(\hat{Y} = 1|D = group2)}{P(\hat{Y} = 1|D = group1)} \quad (3)$$

A.3 Average Odds Difference

Average odds difference measures the average of the difference in the false positive rate and the true positive rate for the differing groups. A value of 0 indicates an equality of odds. A value of -1 or 1 indicates maximum possible inequality.

$$AO = \frac{(FPR_{group2} - FPR_{group1}) + (TPR_{group2} - TPR_{group1})}{2} \quad (4)$$

A.4 Equal Opportunity Difference

The equal opportunity difference metric compares the difference in true positive rates between the two groups. A value of 0 indicates equality between groups. A value of -1 or 1 indicates high inequality.

$$EO = TPR_{group2} - TPR_{group1} \quad (5)$$

The Impact of Semester Gaps on Student Grades

Gary M. Weiss, Joseph J. Denham, and Daniel D. Leeds

Computer and Information Science Department

Fordham University, New York, NY

gaweiss@fordham.edu, jdenham2@fordham.edu, dleeds@fordham.edu

ABSTRACT

College students have great flexibility in choosing when they take specific courses. These choices sometimes are constrained by prerequisite requirements, which determines the order in which pairs of courses may be taken. However, even in these cases the student can choose the number of semesters, or gap, between the pairs of courses. In this paper we study the impact that this gap has on student learning, as measured by course grades. Our methodology accounts for differences in instructor grading policies and in student ability as measured by overall grade point average. Our results can be used to inform course selection and advising strategies. Our study is applied to eight years of undergraduate course data that spans all departments in a large university. Due to space limitations, in this paper we focus our analysis on the semester gaps in Computer Science courses and in Spanish courses. Our results do not show a consistent negative impact on increasing semester gaps between all pairs of courses in a department; however, a negative impact is shown when the gap increases between courses that have a particularly strong relationship and overlapping content.

Keywords

Course sequencing, Student performance, Educational Data Mining

1. INTRODUCTION

Undergraduate college students have considerable flexibility in when they can take specific courses. Course sequencing is typically determined by certain courses being deemed as prerequisites for others. However, even in such cases, there is no requirement that the prerequisite must be taken in the prior semester. Thus there may be a “gap” between when the two courses are taken. A gap may also exist between courses that have no prerequisite relationship. In this study we compute, for all possible course pairs, the impact that this gap has on the course that is taken second. We ignore course pairs with an insufficient number of students in common and focus our analysis on course pairs with a clear connection and overlapping topics.

Our study is based on course grade data from Fordham University, a large university [2] with approximately 15,000 undergraduate and graduate students. As discussed in detail in Section 2, we account for some important factors that could distort our results, such as whether a particular instructor is an easy or hard grader, and whether a course section happens to have a stronger or weaker set of students, based on student overall grade point average (GPA). We introduce a general set of metrics that can be used to assess the

G. Weiss, J. Denham, and D. Leeds. The impact of semester gaps on student grades. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 612–615, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853153>

impact of semester gaps. Our findings indicate that when courses are very strongly related and cover overlapping topics, that an increasing semester gap degrades student performance. However, our results do not show this pattern over the broader set of courses, even if the courses are offered by the same department. The information that we compute can be used to inform the course selection process and to improve departmental advising strategies.

Our study is a special case of the more general work on course sequencing, especially the work on course sequencing as it impacts student grades. Work on the more general area of course sequencing is quite limited and has often been restricted to a single discipline, such as communications [5] or psychology [1]. A prior study of ours did span all academic disciplines in a university, and compared student grades between pairs of courses when the courses are taken in the two possible orderings [3]. Our current study is similar in that we also study only course pairs, but rather than comparing the two sequential orderings, we evaluate the impact of the number of semesters between the courses.

2. COURSE-GRADE DATA SET

The initial data set includes course enrollment and grade data for each undergraduate student in the university. Each individual record corresponds to a specific student taking a specific course (identified by the course department, course number and title, section number, and semester and year) and also has the earned student grade in the course. Grades are represented using a 4-point scale, where a 0 represents an “F” and 4.0 represents an “A.” Our ability to identify individual course sections is important because, as discussed in Section 3.2, student grades are normalized at the section level to account for differences in instructor grading schemes. The data set contains 473,527 records that cover 24,969 students. Due to strict privacy laws, we are unable to share our data set even though the student identifiers have been anonymized.

Our study is focused on student grades as well as the timing between courses. Before proceeding, we need to acknowledge that the course gaps have additional implications. For example, if a large gap exists, then the student will be more advanced in their education (and age) when taking the second course. This by itself can impact student learning. Similarly, the courses that tend to have larger gaps are more likely to have a higher course level, designated by the first digit of the course number (where 1000 represents a first year freshman course and 4000 represents a fourth year senior class). However, this numbering is only a rough guideline and there are even cases where a freshman takes a 4000 level course.

Table 1 shows how “Student Year” and “Course Level” impact a grade, as these factors can influence grade patterns tied to semester gaps. Table 1 shows the average grade by student year (freshman to senior) for each course level and then averaged over all course levels. The final column shows that the average grade increases monotonically with student year; the final row shows that the average grade also increases monotonically with course level. The

number of students in each student year that take courses at a particular course level varies greatly. We track these numbers, although they are not included in Table 1, and use them to compute the displayed averages, which are weighted by the number of students corresponding to each entry. If our results show that an increasing gap degrades student performance, this effect would run counter to the general trend of higher grades as one advances in one's college career.

Table 1. Average grade by year and course level

Student Year	Course Level				Average
	1000	2000	3000	4000	
Freshman	3.108	3.271	3.034	3.209	3.120
Sophomore	3.189	3.281	3.215	3.244	3.222
Junior	3.167	3.310	3.278	3.305	3.261
Senior	3.182	3.342	3.327	3.385	3.320
Average	3.136	3.301	3.271	3.364	

3. CALCULATING GAP PERFORMANCE

In this section we describe how gap performance is measured. Our methodology and the associated metrics incorporate two types of grade normalization, discussed below.

3.1 Basic Methodology

We measure gap performance for potentially every pair of undergraduate courses. For each pair (Course A, Course B), we consider both orderings (i.e., placing Course A first and then placing Course B first). For each ordering, we further partition the students based on the number of semesters between the pair of courses. If the number of students in a partition associated with a particular gap size is at least *gap-threshold*, we calculate the average grade of the students in the second course. If the number of students is below this threshold, the partition is omitted since any average grade may not be reliable. We then compare average grades for different gap lengths to detect the relative impact of larger semester gaps.

For this study we omit summer courses and consider only fall and spring semester courses – we consider the gap between a fall 2014 and spring 2015 course to be the same as the gap between a spring 2015 and fall 2015 course. We do this because summer courses are taken with relative infrequency, and because summer courses compress the material from a fifteen week fall or spring semester into a five week summer session. This compressed schedule, even though it may maintain the number of lecture hours, is likely to have a deleterious impact on student learning. We consider courses taken in consecutive semesters to have a gap of 1 (i.e., one semester difference), although we recognize it might be more natural to consider this as a gap of 0. However, using our scheme courses taken in the same semester can be represented with a gap of 0; we ignore such cases from our present analysis but may consider them in the future. Taking courses simultaneously is very different from taking courses sequentially, and merits a separate analysis.

3.2 Normalizations

We normalize our student grades to account for two confounding factors. We first normalize grades at the course section level to account for instructors that are easy versus harsh graders; we then normalize for student ability, as measured by the student's grades across all of their courses (i.e., their GPA). We apply these sequentially, so we never normalize only for student ability. We refer to the resulting normalized student grades as *instructor* normalized and *compound* (instructor and student ability) normalized grades. In order to provide more succinct names and table headings, and to reflect that these normalizations are applied sequentially, we refer

to these as *Level 1* and *Level 2* normalizations. These two levels of normalization have been used in several of our labs other studies, including the previously discussed study on the impact of course pair orderings on student grades [3].

The Level 1 instructor normalization takes the raw student grades in an individual course section and applies z-score normalization. This will account for instructors who assign grades with higher, or lower, overall means, so that these factors do not obscure the semester gap effects. Equation 1 describes the normalization process. In this equation, $L1_i$ represents the L1 instructor normalized grade of student i in the course section, x_i represents the initial grade of the student, μ represents the mean grade of the course section, and σ represents the standard deviation of the course section grades.

$$L1_i = (x_i - \mu) / \sigma \quad (1)$$

The next normalization accounts for student ability. That is, a particular section may contain students who are stronger-than-usual overall. While such affects may be expected to average out, they nonetheless could distort our results. We measure student ability by the student's grades in all of their courses. We could try to do better by using a student's major GPA for major courses, humanities GPA for humanities courses, and so on, but we do not want to overly complicate the calculations. This step takes the normalized scores produced by Equation 1 and then applies another round of z-score normalization, as described by Equation 2. In this equation, $L2_i$ represents the Level 2 compound normalized grade of student i in the course section, $L1_i$ represents the L1 instructor normalized grade of student i , μ_i represents the mean instructor normalized grade across all courses taken by student i (i.e., instructor normalized GPA), and σ_i represents the standard deviation of this student's instructor normalized grades.

$$L2_i = (L1_i - \mu_i) / \sigma_i \quad (2)$$

These steps are done at the course section level. These values are then aggregated over all course sections for each course. These are the values that we report in our results as long as there are a sufficient number of values (i.e., students). As mentioned in Section 3.1, our metrics are based on the students who take the second course a specific number of semesters after the first course. We refer to the metrics computed over these partitions of student grades as the L1 gap and L2 gap metrics, where each of these metrics in turn is associated with a specific gap length (e.g., we can talk about the L1 gap=1 value versus the L1 gap=2 value). In our discussion we prioritize the L2 gap metric over the L1 gap metric.

3.3 Example Gap Performance Entry

An example is provided in Table 2 to provide a better understanding of the metrics that were just introduced. The table includes just a single entry, which corresponds to "Introductory Biology I" followed by "General Chemistry 1." In our dataset there are a total of 1301 students who took these two courses in this sequence. The number of students who took it with a gap of 1 and a gap of 2 is not provided in the table but is above *gap-thresh* (the results in Section 4 use *gap-thresh*=40 unless otherwise specified). Table 2 only includes the Level 2 results. The value for students taking "Gen Chem 1" in the semester right after taking "Intro Bio 1" is 0.519, while the value for those waiting an extra semester (so they take it the same semester in the following year) is -0.362. Since the value decreases, the delay hurts student performance. The difference between these values (gap 1 minus gap 2) is 0.881 and is denoted as the L2 difference (L2 Diff). We generally do not show the gap values past a gap of 2 because there are often insufficient numbers of students to satisfy *gap-thresh*. For this reason we focus on the

difference between a gap of 1 and 2, although we provide one example with a larger gap in Table 3. As we accumulate additional student data, we may be able to extend our analyses to look at larger gap lengths.

Table 2. Sample course gap table entry

Course 1	Course 2	Corr.	Students	Level 2 Gap		L2 Diff (gap 1-2)
				1	2	
Intro Biol	Gen Chem1	0.64	1301	0.519	-0.362	0.881

We tend to focus our analyses on courses that are normally taken closely together—if this is not the case then the courses are most likely not highly linked, and a delay is less likely to matter. Table 2 specifies the Pearson correlation (Corr.) between the student grades in the two courses. This correlation, which is computed independent of order, tends to identify courses that are highly related/linked. The link may not be causal—it could just be that the courses utilize similar skills so that students who perform well in one perform well in another. But we do feel that a high grade correlation is a necessary, but not sufficient, condition for two courses to have their grades causally linked—and hence have the possibility of a semester gap having a substantial influence on student performance. Thus we can use grade correlation to help guide our search for courses to focus our attention on. This use of grade correlation was studied in one of our prior studies [4], which suggested that a grade correlation of 0.5 or higher suggests that courses are linked.

Table 3 includes the only entry in this paper with a gap length greater than 2. The example involves “General Chemistry 1” followed by “General Chemistry 2.” This example was selected by sorting all of our results by decreasing levels of grade correlation and then selecting the first entry. Note that the performance degrades when the gap transitions from 1 to 2. This trend does not continue to a gap length of 3, but student performance at gap length of 3 is still below that of gap length of 1. The entries for gap length of 3 may be less reliable since they involve fewer students, although they still exceed the *gap-threshold* of 40.

Table 3. Course gap entry for GenChem1 → GenChem2

Metric	Corr.	Students	Gap Length		
			1	2	3
L1	0.730	1328	-0.040	-0.753	-0.615
L2	0.730	1328	-0.337	-1.012	-0.689

4. MAIN RESULTS

This section covers our main results. Section 4.1 take a detailed look at Computer Science courses and then Section 4.2 looks at Spanish courses. We study Computer Science courses because all of the authors of this study are affiliated with a Computer Science department. Foreign language courses were selected since our intuition is that courses in a foreign language will be heavily sequenced and the performance in one course will impact the performance of subsequent courses; Spanish courses were selected because Spanish is the most popular foreign language studied in our university.

4.1 Computer Science Results

In this section we take a detailed look at the course gap results for Computer Science (CS) courses. In order to provide a fair analysis, all course pairs that satisfy a *gap-threshold* of 40 are displayed in Table 4.

Before analyzing the results, it is important to understand the relationship between CS courses, since the gap values are not as likely to be meaningful for courses that are not highly related. Thus we

first highlight the course pairs where the courses cover the same topic(s), although perhaps at different levels. In these cases the first course will be an *essential* prerequisite for the second course. Our introductory programming sequence is CS1→CS2→Data Structures, and each course builds on the prior ones and utilizes the same programming language. While other courses may be related by a prerequisite, the prerequisite may not be as essential. For example, Databases has CS1 as a prerequisite, but mainly just to ensure that the student has had a general exposure to programming. The only other course pair in Table 4 that may share an “essential” relationship is “Data Structures” and “Algorithms,” although there are substantial differences, as the first course has a heavy programming component, while the second course generally does not.

Table 4. Computer Science course gap table

Course 1	Course 2	L1 Diff	L2 Diff	Corr.	Stdnts	L1 gap		L2 gap	
						1	2	1	2
CS1	CS2	.250	.308	.568	582	.052	-.198	.078	-.730
CS2	DataStruct	.301	.357	.549	351	.072	-.229	.063	-.293
Databases	DataComm	.044	.346	.526	170	.000	-.044	.452	.107
Databases	OS	-.017	.069	.520	187	.146	.163	.188	.118
CS1	Databases	.116	.024	.443	274	-.193	-.309	-.087	-.111
CS2	Databases	-.029	-.009	.469	238	.069	.098	.304	.313
DiscMath	CS1	-.196	-.137	.382	300	-.320	-.125	-.160	-.024
DataStruct	Algrthms	-.163	-.346	.526	232	.011	.174	-.131	.215
DataStruct	OS	-.168	-.353	.550	226	-.114	.054	-.284	.069
DataStruct	TOC	-.143	-.412	.460	251	-.149	-.006	-.402	.010
CS1	CompOrg	-.292	-.446	.410	268	-.238	.053	-.151	.296
CS2	CompOrg	-.239	-.533	.463	256	.101	.340	.166	.699
CS2	DataMining	-.129	-.821	.492	212	-.070	.059	-.176	.646

The key observation from Table 4 is that the gap between CS1 and CS2, and the gap between CS2 and Data Structures, has a large L2 difference—and in fact they have the two largest positive differences. The same pattern is also observed with the L1 normalized differences. The grade correlation between students who take CS1 and CS2 (0.568), and CS2 and Data Structures (0.549), is quite high, and in fact are higher than for just about all other course pairs. This further supports the strong relationship between these pairs of courses. In other cases, when the grade correlation is high it could just be that the courses rely on similar student abilities and there is no causal link. Nonetheless, we generally would expect a relationship between grade correlation and L2 Difference, even if some of the relationships are not causal. The scatter plot and associated trendline in Figure 1 demonstrates that there is such a relationship, and the Pearson correlation coefficient between these two measures of 0.459 confirms this (this is considered to be a moderate positive correlation).

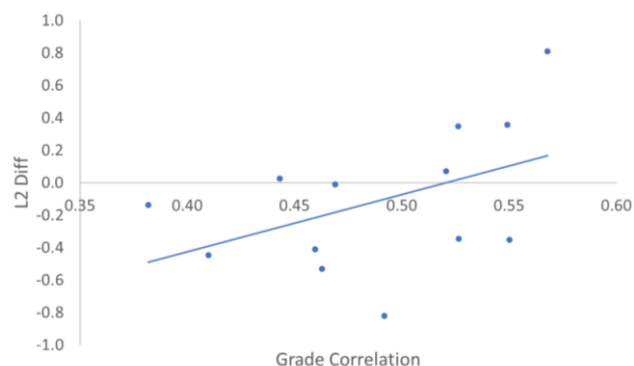


Figure 1. Grade correlation versus L2 diff (Computer Science)

The remaining entries in Table 4 do not support the hypothesis that a larger gap will yield poorer performance, since in many cases the L1 and L2 differences are negative. This suggests that the gap has no substantive negative impact on student learning in those cases. Often negative differences are associated with first-courses intended to be taken early in the major (CS1, CS2, and Data Structures) and second-courses intended to be taken in the middle years of the major with relatively few official prerequisites. Perhaps the gap poses a benefit in these cases by allowing the students time to reflect and grow through partially-tangential academic experiences before joining the second course. It should be noted that we already advise students to take our introductory programming sequence CS1 → CS2 → Data Structures as quickly as possible, but our universities' large liberal-arts core curriculum, which includes well over a dozen courses, sometimes interferes with this advice.

4.2 Spanish Results

The next discipline/department that we look at is Spanish, for the reasons provided earlier. Students who do not major in a science are all required to take a substantial amount of a foreign language—beyond the placement credits that most students receive. Our university catalog reveals the following prerequisite structure for the Spanish courses that appear in Table 5:

Spanish 1 → Spanish 2 → Language and Literature → Approaches to Literature → Latin America: Literature and Culture

The first two courses in the sequence are standard intermediate-level courses that focus on language fundamentals such as reading, writing, listening, and speaking. They do require two introductory Spanish courses, but most students receive placement credit for those based on their high school foreign language requirement. The Spanish Language and Literature course introduces Spanish literature, but a key focus in the course is using these topics to review the language fundamentals. The last two courses assume language competence and the focus goes beyond language fundamentals. Based on this background, one would expect that “Spanish 1” and “Spanish 2” would have the strongest relationship, followed by “Spanish 2” and “Language & Literature.”

Table 5. Spanish course gap table

Course 1	Course 2	L1	L2	Corr.	Stdnts	L1 gap		L2 gap	
		Diff	Diff			1	2	1	2
Spanish1	Spanish2	.686	.420	.663	2348	.001	-.685	-.036	-.456
Spanish2	Lang&Lit	.471	.232	.640	2472	-.023	-.494	-.236	-.468
ApprToLit	LatinAmerica	-.111	.151	.592	166	-.092	.019	-.451	-.602
Spanish1	Lang&Lit	-.317	-.246	.583	2020	-.395	-.078	-.580	-.334
Lang&Lit	ApprToLit	-.113	-.397	.390	496	.009	.122	-.747	-.350

The data in Table 5 agrees with the expectation that the courses that are most related will most negatively impacted by an extra semester gap. Those courses also have the highest grade correlation, which further suggests a close relationship.

5. CONCLUSION

This paper introduced a methodology for evaluating the impact of semester gaps between courses on student performance. Performance is measured based on student grades and two levels of normalization are used to account for differences in instructor grading policies and in student ability. This methodology was applied to eight years of undergraduate course data. This initial study focused on the impact of semester gaps on Computer Science courses and on Spanish courses; the former was selected to utilize the

domain knowledge of the authors while the latter was selected due to the expected strong impact between introductory foreign language courses. In both cases a longer gap did not consistently degrade student performance. This may be explained by the fact that even within a given discipline, the knowledge gained in one course may not necessarily have a strong impact on the performance in the other course. However, for both Computer Science and Spanish, the courses with the clearest direct connections *did* show a degradation in performance with a longer gap. In these two cases the courses with the largest such degradation were introductory courses where both pairs of courses *share the same name*, but with a different suffix (i.e., “Computer Science 1” and “Computer Science 2” and “Intermediate Spanish 1” and “Intermediate Spanish 2”). We believe that this connection is not accidental and that performance in highly related courses suffer from degradation in retained knowledge over time, as measured in the present work. We further note that if two courses have no meaningful link in grades, a larger gap should lead to *improved* performance in the second course based on our results in Table 1, which show that students earn higher grades the further along they are in their academic career. Our work has the potential to improve academic advising in that an advisor, or an automated tool, could warn students when a delay is most likely to lead to degraded performance.

This study has several limitations that will be addressed in future work. Although our university is considered a large university by the Carnegie classification [2] since it has more than 10,000 degree seeking students across all levels, we often found that the number of students taking pairs of related courses was often quite modest, and the number quickly diminished with the number of intervening semesters. Our immediate plan is to obtain several years of additional data, since our data is currently three years old. Longer term, we hope to obtain data from another university, or to lend our analysis tool to other researchers, who can apply it to their university data. We also plan to form higher level hypotheses about the role of time between courses, and combine it with domain knowledge across many disciplines, so that we can generate more reliable conclusions. For example, if we had a list of the two most highly related courses in each academic major, we could compare the impact of gap length on these collective pairs of courses.

6. REFERENCES

- [1] Betancur, L., Rottman, B. M., Votruba-Drzal, E., and Schunn, C. 2019. Analytical assessment of course sequencing: The case of methodological courses in psychology. *Journal of Educational Psychology*, 111(1), 91.
- [2] The Carnegie Classification of Institutions of Higher Education. <https://carnegieclassifications.iu.edu/>.
- [3] Gutenbrunner, T., Leeds, D.D., Ross, S., Riad-Zaky, M., and Weiss, G.M. 2021. Measuring the academic impact of course sequencing using student grade data. In *Proceedings of The 14th International Conference on Educational Data Mining*, International Educational Data Mining Society, Paris France, June 29-July 2, 799-803.
- [4] Leeds, D.D., Zhang, T., and Weiss, G.M. 2021. Mining course groupings using academic performance. In *Proceedings of The 14th International Conference on Educational Data Mining*, International Educational Data Mining Society, Paris France, June 29-July 2, 804-808.
- [5] Richards, A. S. 2012. Course sequencing in the communication curriculum: A case study. *Communication Education*, 61(4), 395-427. DOI= 10.1080/03634523.2012.713500

Assessing Instructor Effectiveness Based on Future Student Performance

Gary M. Weiss, Erik Brown, Michael Riad-Zaky, Ruby E. Iannone and Daniel D. Leeds
Computer and Information Science Department
Fordham University, New York, NY
{gaweiss, ebrown92, mriadzaky, riannone, dleeds}@fordham.edu

ABSTRACT

Educational institutions rely on instructor assessment to determine course assignments, which instructors to retain or promote, and whom to provide with additional assistance or training. Instructor assessment is most commonly based on student surveys or peer evaluation—which are both subject to the evaluator’s personal biases. This study describes an assessment method based on future student grade performance, which has the potential to avoid these biases. This study is based on eight years of undergraduate course-grade data from over 24,000 students in a large metropolitan university. The methodology introduced in this paper accounts for confounding factors, such as diverse instructor grading policies and varying student abilities. Top and bottom performing instructors are identified for each course.

Keywords

Instructor effectiveness, instructor evaluation, data mining, educational data mining, grade analysis, data analysis.

1. INTRODUCTION

Assessing instructor effectiveness is important for determining which instructors to retain or promote, optimal assignment of courses, and providing additional mentorship or training to weak instructors [3, 8]. It is also often a key factor in tenure decisions. In a university, these assessments are typically done through student surveys or peer evaluations based on classroom observations [8]. Both of these methods are subject to the biases of the evaluators, which may be impacted by instructor gender and race, and may not measure student learning [1, 2, 5, 7]. The justification for using student surveys is derived from several studies in which positive correlations are found between student evaluations and instructor effectiveness as measured through exams at the end of each course. However, a recent meta-analysis conducted on thirty-two of these studies shows that there is no such positive correlation for the studies containing the most course sections, indicating earlier conclusions were due to a lack of data and providing argument against the use of student evaluations to measure instructor effectiveness [7]. These studies also measure instructor effectiveness using the grades for the course being taught. Given that the exams and grades are usually designed by the instructor, this yields another potential source of bias; our methods avoid this bias by relying on students’ performance in future courses. Peer evaluations are most likely subject to similar biases.

G. Weiss, E. Brown, M. Riad-Zaky, R. Iannone, and D. Leeds. Assessing instructor effectiveness based on future student performance. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 616–620, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852940>

The method introduced in this paper assesses instructors by quantifying their impact on future student grade performance. If students with a given instructor perform better (worse) in future courses than students who have a different instructor, then the instructor is ranked favorably (unfavorably). Our study only assumes basic course-grade data is available, and does not account for all potential confounding factors, such as the time of day of the class or class size [9]. However, we do account for grade-related factors such as instructor grading leniency and student ability as measured by grade-point average. We have developed a publicly available Python-based software tool that implements our methodology and generates the instructor effectiveness metrics [6].

Our study identifies instructors who appear to be much more or less effective than other instructors based on their student’s future performance. Our analysis first focuses on two case studies, assessing instructors of “Spanish 1” and “Computer Science 2” based on future performance in “Spanish 2” and “Data Structures,” respectively. We then identify the best instructors in the university based on the teaching of a single course, and then identify the top-10 and bottom-10 instructors based on the performance over all courses each instructor teaches, with future student performance measured on all future courses taken within a single department. We discuss several interesting patterns across these results.

2. STUDENT COURSE-GRADE DATA SET

The work presented in this paper is based on eight years of undergraduate course-grade data from Fordham University. Each record in the dataset represents a student earning a grade in a specific course section and includes the following fields: student identifier, instructor ID, course name, course number, course department, course term (semester and year), and student grade using a 0.0 (F) - 4.0 (A) scale. Table 1 provides key dataset statistics. In order to enhance privacy, student identifiers were remapped and data for course sections with fewer than five students were omitted. Even with such measures, due to federal regulations we are not permitted to publicly share the dataset.

Table 1. Summary Dataset Statistics

Feature	Unique Values
Record Number	442,230
Student ID (SID)	24,654
Instructor ID (IID)	2,195
Course Name & Number	2,505
Course Section	21,504

Demographic information was not included in the data set as it could facilitate de-anonymization. We therefore characterize the population using university statistics for the middle year of the data: gender distribution is 60% female and 40% male, and the racial/ethnic breakdown is 55% White, 14% Hispanic, 11% Asian, 7% International, 4% Black, and 9% other. The majority of students are between the ages of 17 and 22.

3. MEASURING INSTRUCTOR BENEFIT

This section describes the methodology used to calculate instructor benefit and introduces our three instructor benefit metrics. This methodology is implemented in a publicly available Python-based software tool developed by our research group [6], which enables other researchers to apply our research to other student grade datasets. The steps used to generate our results are summarized in Figure 1 and described in subsequent subsections.

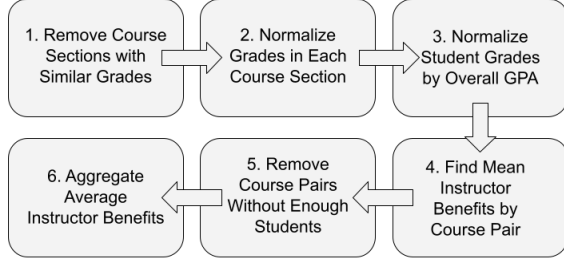


Figure 1. Overview of Data Processing Steps

3.1 Remove Sections with Too Similar Grades

Differences in instructor effectiveness can only be measured if there is reasonable variance in the grades assigned to students. Thus we remove the data associated with course sections with very low grade variance. Step 1 computes the standard deviation of the grades in every course section and eliminates sections where the standard deviation is below $MinSD$. The distribution of grade standard-deviation values at the section level is provided in the Appendix, and Section 4 specifies the $MinSD$ value of 0.2 that is used throughout this study.

3.2 Normalize Grades in each Section

To account for different instructor grading schemes, we employ z-score normalization for each course section. The Level-1 normalized score, defined below, tells us how many standard deviations away from the course section mean a student scores.

$$L1_{SID}^{CrsSec} = \frac{G_{SID}^{CrsSec} - \mu_{CrsSec}}{\sigma_{CrsSec}}$$

In this formula, $L1_{SID}^{CrsSec}$ is the normalized course grade for student SID in the specified course section $CrsSec$, G_{SID}^{CrsSec} is the original grade for student SID in $CrsSec$, and μ_{CrsSec} and σ_{CrsSec} are the mean and standard deviation of the grades in $CrsSec$. Instructor benefits calculated using this Level-1 normalized grade are referred to as L1 Instructor Benefits. Instructor benefits calculated using the unnormalized grades are referred to as Grade Benefits (although they can be viewed as Level-0 Instructor Benefits).

3.3 Normalize Grades by Overall Performance

Student performance is not just dependent on the effectiveness of the instructor but also depends on a student's abilities. We therefore employ a second level of grade normalization that is based on the student's overall performance in all of their courses (i.e., GPA). Without this normalization, an instructor that coincidentally is assigned high-performing students will appear to perform better than an instructor that is assigned weaker students. This Level-2 normalization is defined by the formula below and L2 Instructor Benefits are calculated using these values.

$$L2_{SID}^{CrsSec} = \frac{L1_{SID}^{CrsSec} - \mu_{SID}^{norm}}{\sigma_{SID}^{norm}}$$

In this formula, $L2_{SID}^{CrsSec}$ is the Level-2 normalized grade of student SID in course section $CrsSec$, $L1_{SID}^{CrsSec}$ is the Level-1 normalized score from the prior step, and μ_{SID}^{norm} and σ_{SID}^{norm} are the mean and standard deviation of student SID 's L1-normalized grades across all courses.

3.4 Find Instructor Benefit by Course Pair

We next consider every ordered course pair, $C1 \rightarrow C2$, where course $C1$ is taken prior to course $C2$. Assume that the instructor for $C1$ has an instructor ID, IID . The instructor benefit associated with instructor IID teaching $C1$ based on $C2$ performance is computed using the $C2$ grades for those students who previously had instructor IID for $C1$. The type of $C2$ grade (unnormalized, L1-normalized, L2-normalized) determines the type of Instructor Benefit (Grade, L1, L2). The calculations just described are aggregated over all sections for a given course. More formally, $IB_{IID}^{C1 \rightarrow C2}$ is the instructor benefit (IB) for students taking $C2$ after taking $C1$ with instructor IID .

$$IB_{IID}^{C1 \rightarrow C2} = \langle GRADE_{SID \in (C1, IID)}^{CrsSec \in (C2)} \rangle$$

In this formula, $CrsSec \in (C2)$ is any course section of course $C2$, $SID \in (C1, IID)$ is every student who took course $C1$ with instructor IID , and $\langle x \rangle$ is the average of all values in x .

$IB_{IID}^{C1 \rightarrow C2}$ is computed for all ordered course pairs $\langle C1, C2 \rangle$ and for every $C1$ instructor, as long as at least 80% of the students who complete both courses take $C1$ first. This restriction ensures that we only evaluate instructor effectiveness between courses that are taken in the expected order. In Section 3.6, we discuss aggregating the instructor effectiveness metrics for each instructor teaching $C1$ over all $C2$ courses, and then again over all $C1$ courses for each instructor. However, we believe that these higher level metrics are less meaningful, since each $C1$ course is often best evaluated on a single $C2$ course (e.g., Spanish 1 is best evaluated using performance in Spanish 2). For this reason most of the results in Section 5 are at course-pair level. When the most appropriate future course is not clear based on domain knowledge, the choice can be guided by looking at the course pairs with the highest pairwise grade correlation, as described in one of our recent studies [4].

3.5 Remove Course Pairs with Few Students

Instructor effectiveness is computed for pairs $C1 \rightarrow C2$. For the resulting instructor effectiveness metrics to be reliable, the instructor will need to have taught at least $MinStudents$ in $C1$ who subsequently completed $C2$. However, since this is a comparative statistic, we also require that there are $MinStudents$ who completed $C1$ with *other* instructors and then completed $C2$ (hence if $MinStudents = 50$ there must be at least 100 students that took $C1$ and then completed $C2$). Section 4 explores how the $MinStudents$ threshold impacts the number of available course pairs.

3.6 Aggregate Average Instructor Benefits

Course-pair level instructor benefit metrics are aggregated to yield higher level views of instructor performance. The instructor benefit values for each $\langle C1, C2 \rangle$ course pair are first aggregated over the set of $C2$; restrictions on $C1$ and $C2$ may be applied. The results in Section 5.3 are at the instructor level but are aggregated over all $C1$ courses taught by instructor IID in a single department and measured on future performance over classes in another department (i.e., $AIB_{IID}^{D1 \rightarrow D2}$). This aggregation formula, which is provided below, also weights the courses by the number of students.

$$AIB_{IID}^{D1 \rightarrow D2} = \frac{1}{\sum_{C1 \in CX_{IID}} |\{S \in CX_{IID}\}|} \sum_{\substack{(C1 \in D1) \\ (C2 \in D2)}} IB_{IID}^{C1 \rightarrow C2} |\{S \in IB_{IID}^{C1 \rightarrow C2}\}|$$

The computation is performed across all relevant course pairs in the department $C1 \rightarrow C2$, where $IB_{IID}^{C1 \rightarrow C2}$ is an instructor benefit from $C1$ to $C2$, $|\{S \in IB_{IID}^{C1 \rightarrow C2}\}|$ is the number of students who took the instructor in that particular course pairing, and $\sum_{CX, IID} |\{S \in CX_{IID}\}|$ is the number of students who took a course with instructor IID (summed over all possible courses CX). The formula above can be used to find, amongst other things, the instructor effectiveness scores for a Computer Science instructor when evaluated on future Math classes, as well as when evaluated on future Computer Science classes (i.e., when $D1 = D2$).

4. Threshold Sensitivity Analysis

We selected appropriate thresholds for $MinSD$, the minimum standard deviation of section grades, and $MinStudents$, the minimum number of students in each course pair. The Appendix provides additional relevant information related to the selection of these thresholds. We select $MinSD=0.2$, retaining most (20,904 of 21,504) course sections while ensuring some variability in student grades, and $MinStudents=50$, to keep a large number of course pairs while maintaining reliability of the instructor benefit metrics.

5. RESULTS

This section provides our instructor benefit results. Section 5.1 and Section 5.2 provide results at the course-pair level, while Section 5.3 identifies the top and bottom performing instructors based on their performance across all courses in a department. Grade Benefit, Level-1 Benefit, and Level-2 Benefit metrics are all provided, with a focus on Level-2 Instructor Benefits.

5.1 Section Level Course-Pair Results

This section provides results at the course-pair level. Due to space limitations we can only provide instructor benefit results for instructors teaching Spanish 1 based on future student performance in Spanish 2 and for instructors teaching Computer Science 2 (CS2) based on future performance in Data Structures. We include the Spanish courses because they are popular and Computer Science courses because they are offered by our home department and there is great interest in Computer Science education. In both cases, the course pairs are part of a common introductory sequence, and the second course directly follows the first course.

Table 2. Instructor Benefit for Spanish 1 \rightarrow Spanish 2

Instructor ID	Sections Taught	Total # Students		Instructor Benefit		
		Spanish 1	Spanish 2	Grade	Level 1	Level 2
F980	22	381	325	-0.024	0.016	0.050
F494	17	367	295	-0.189	-0.091	-0.090
F787	12	217	166	-0.154	-0.079	-0.210
F424	11	231	191	-0.278	-0.259	-0.146
F425	11	213	171	0.034	-0.076	-0.097
F819	10	201	176	0.050	0.124	0.233
F883	9	154	129	0.030	-0.039	0.065
F719	8	179	134	0.090	0.076	0.147
F890	7	172	138	-0.097	-0.009	-0.282
F541	7	86	67	0.061	0.045	-0.219
All	189	3485	2644	-0.088	-0.092	-0.073

The results for the Spanish classes are summarized in Table 2. Sections where fewer than five students continue to Spanish 2 are excluded to allow section-level instructor benefit values to be reliable for statistical analyses. Due to space limitations, instructors with fewer than seven sections are not listed but are included in the summary statistics in the last row.

Our analysis focuses on Level 2 instructor benefit because it accounts for the two confounding factors discussed earlier, but Figure 2 shows that the three metrics are generally correlated, and

the Pearson correlation coefficient (ρ) using the 189 section level values confirms this with $\rho=0.74$ between the Grade and Level-1 metric and $\rho=0.76$ between the Level-1 and Level-2 metric.

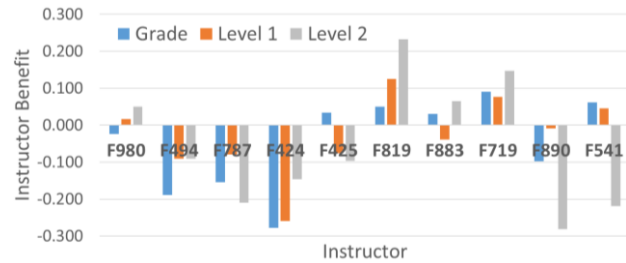


Figure 2. Instructor Benefit Metrics for Spanish 1 Instructors

Table 2 and Figure 2 show that there are substantial differences in the Level-2 values since they vary from +0.233 to -0.282. Small differences in instructor ability may be hard to distinguish, so in this initial study, we focus on cases with the largest differences and where the values are based on many students. Given this, we conclude that instructor F819 is highly effective, while F890, F541, and F787 are least effective. A two-sample unequal variance t-test at the section level for Instructor F819 (+0.283) and F787 (-0.210) yields $p=0.0156$ for the one-tailed distribution and $p=0.0312$ for the two-tailed distribution. These p-values suggest that the differences are statistically significant, although this is partially due to comparing instructors at the two extremes. More students would make more refined assessments possible.

Table 3 provides analogous results for the <CS2, Data Structures> course pair. Instructor data is limited because the CS major was not heavily populated in the timeframe considered (2010 to 2018). The table includes all instructors that taught three or more sections of CS2 (the bottom two instructors did not meet our preferred $MinStudents$ threshold of 50). Based on the level 2 instructor benefit, two instructors are strongly positive and two moderately negative. A t-test on the first two instructors in Table 3 yields p-values of 0.00030 (1-tail) and 0.0003 (2-tail).

Table 3. Instructor Benefit for CS2 \rightarrow Data Structures

Instructor ID	Sections Taught	Total # Students		Instructor Benefit		
		CS2	DataStr.	Grade	Level 1	Level 2
F212	12	293	158	-0.304	-0.226	-0.189
F177	4	92	62	0.237	0.151	0.396
F589	3	56	36	-0.329	-0.177	-0.228
F653	3	35	33	-0.385	-0.042	0.400
All	32	697	410	-0.227	-0.145	-0.054

The section level results suggest that it is possible to distinguish between high and low-performing instructors when using future student performance in a single highly related course. It may be difficult to reliably assess less extreme differences, but universities with larger classes or higher teaching loads should be better able to perform more refined assessments.

5.2 Global Course-Pair Instructor Results

Table 4 provides the best Level 2 Instructor Benefit results at the course-pair level. The course pairs are restricted to the same department or between departments that share major requirements, since it is best to measure instructor effectiveness using related courses. The results are based on $MinSD=0.2$ and $MinStudents=50$. Each entry in Table 4 corresponds to a single instructor. Course names are abbreviated using department codes and course numbers, but the full names are provided in our discussion.

Table 4. Top 6 Instructor Course Pairs by Level 2 Benefit

Course1	Course2	Grade Benefit	Level 1 Benefit	Level 2 Benefit
Chem 211	Bio 342	0.11	0.09	1.50
Econ 220	Econ 332	0.53	0.50	1.12
Phys 140	Chem 121	0.24	0.43	0.79
Chem 121	Chem 212	0.14	0.22	0.76
NatSci 304	NatSci 321	0.44	0.48	0.75
Comm 112	Comm 242	0.41	0.38	0.72

The courses pairs that appear in Table 4 exhibit a strong relationship between Course 1 and Course 2. For example, the first row involves “Organic Chemistry Lab” and “Biochemistry,” while the two Natural Science courses correspond to “Organic Chemistry I Lab” and “Organic Chemistry II.” Our belief is that the instructor associated with each entry is a very effective instructor, although, as discussed in Section 6, we cannot validate this. The Grade and L1 Benefit values are rarely as high as the L2 Benefit values, which indicates that the second round of normalization has a substantial impact. Note that students who take Econ 220, with the instructor represented by that entry, obtain a grade that is, on average, 0.53 higher than otherwise expected; this corresponds to a difference of more than a half letter grade.

5.3 Department Level Instructor Results

This section describes the aggregate effectiveness of an instructor ($AIB_{ID}^{D1 \rightarrow D2}$, as defined in Section 3.6.) based on all courses that instructor teaches in one department, measured by student success in future courses in one (potentially different) department. We consider these results to be less meaningful than results on course-pairs selected for mutual relevance. Nonetheless, some interesting high level observations arise. The results in Table 5 include the top and bottom performing instructors using $MinStudents=50$ and $MinSD=0.2$. Each entry corresponds to a single instructor.

Table 5. Top and Bottom 10 Instructors by Level 2 Benefit

Course 1 Department	Course 2 Department	Instructor Benefit		
		Grade	Level 1	Level 2
Top 10 Instructors				
Political Sci.	Natural Science	0.317 ⁽⁸⁾	0.340 ⁽⁵⁾	0.588
Mathematics	Mathematics	0.033	0.242 ⁽⁸⁾	0.570
Economics	Theology	0.140	0.136	0.546
Economics	Theology	-0.127	-0.035	0.525
Italian	Physics	0.131	0.225	0.510
Art History	Spanish	0.314 ⁽⁹⁾	0.260 ⁽⁶⁾	0.503
Philosophy	Communications	0.296	0.350 ⁽²⁾	0.502
Mathematics	Chemistry	0.063	0.103	0.497
Natural Science	English	0.267	0.237 ⁽⁹⁾	0.488
Physics	Political Science	0.233	0.009	0.465
Bottom 10 Instructors				
Mathematics	Mathematics	0.163	-0.033	-0.585
Mathematics	Mathematics	-0.182	-0.131	-0.563
Physics	Physics	0.154	-0.075	-0.511
Sociology	Physics	0.107	-0.066	-0.476
Natural Science	English	-0.228	-0.203	-0.463
Visual Arts	Anthropology	-0.209	-0.072	-0.422
Natural Science	Natural Science	-0.475 ⁽¹⁾	-0.372 ⁽³⁾	-0.416
Chemistry	Biology	-0.342 ⁽⁸⁾	-0.443 ⁽¹⁾	-0.415
Comp. Science	Physics	0.304	-0.185	-0.411
Mathematics	Natural Science	0.085	-0.042	-0.406

The instructor benefits in Table 5 indicate that there is a substantial difference between the top and bottom performing instructors. Many unnormalized grade differences are about 0.3, representing one-third of a letter grade difference. The values for the three effectiveness metrics appear correlated. To verify this we computed

the Pearson correlation ρ between all three pairs of metrics, with the following results: $\rho(\text{Grade, Level 1}) = 0.977$, $\rho(\text{Grade, Level 2}) = 0.990$, $\rho(\text{Level 1, Level 2}) = 0.989$. These correlations are higher than in the prior section, showing a difference through aggregation over many sections and courses. For comparison, when the Grade or Level 1 benefit metric appears in the top-10 or bottom-10 for the listed instructor entry, we provide the rank in parentheses as a superscript (e.g., the instructor in the first row of data has the third highest Level-1 Benefit). The ranks are not needed for Level 2 Benefits since they are already in rank order.

We generally would expect that instructor effectiveness in a course will have the biggest impact on other courses in the same discipline. Many of the entries do involve the same department or related departments, but quite a few do not. There is much more agreement between the departments for the bottom 10 instructors, seeming to indicate weak instructors fail to convey field-specific concepts for future use, while strong instructors may convey broader skills useful across disciplines. We note that STEM (Science, Technology, Engineering, and Math) instructors account for 80% of the bottom performing instructors but only 40% of the top performing instructors, which is plausible given that STEM graduate programs generally provide little pedagogy instruction.

6. CONCLUSION

Bias in student and peer evaluations of instructor effectiveness have been widely observed [1, 2, 5, 7], supporting the need for more objective assessment methods. This study presents an alternative method for instructor evaluation based on student performance in future courses. Our study accounts for instructor grading leniency and overall student ability; these factors impact assessment, but all three metrics are nonetheless highly correlated.

Instructor assessment appears most appropriate at the course level and provides most insight when considering future performance in a single, highly related course. We focused on instructor performance for Spanish 1 and CS2, based on future student performance in Spanish 2 and Data Structures, respectively. In both cases, instructor benefit varied substantially and the Level 2 instructor benefit for instructors at these two extremes differed with reasonable levels of statistical confidence. Our methodology distinguishes between instructors and identifies high and low performing instructors. Evaluation of single instructors across courses was less clear, but revealed patterns, such as the weakest instructors often being associated with STEM disciplines.

The methodology and metrics described in this paper are calculated from traditional student course-grade data using a publicly available Python-based tool developed by our research group [6]. This tool can be used by other researchers and practitioners to extend our analysis to other educational institutions. We plan to improve the tool’s documentation and usability in the near future.

There are numerous areas for further work. Increasing the size of our data set would substantially strengthen future analysis, especially within our Computer Science department. Effects of instructor rank, title, years of experience, gender, and race also would be valuable to study. Furthermore, we aim to identify further discipline-based patterns, such as differences in instructor effectiveness distributions across departments.

The most fundamental limitation of this work relates to validation. Currently we only perform limited validation across course sections. Additional validation will inherently be limited since there is no way to assess the “ground truth.” Still, we aim to measure the relationship with weaker metrics like student survey results.

7. REFERENCES

- [1] Boring, A., Ottoboni, K., and Stark, P.B. 2016. Student evaluations of teaching (mostly) do not measure teaching effectiveness. *Science Open Research*. DOI=<https://doi.org/10.14293/S2199-1006.1.SOR-EDU.AETBZC.v1> .
- [2] Chávez, K., Mitchell, K. 2020. Exploring bias in student evaluations: Gender, race, and ethnicity. *PS: Political Science & Politics*, 53(2), 270-274. DOI=<https://doi.org/10.1017/S1049096519001744>.
- [3] Goldhaber, Dan. 2015. Teacher effectiveness research and the evolution of U.S. teacher policy. *The George W. Bush Institute*. <https://files.eric.ed.gov/fulltext/ED560206.pdf>.
- [4] Leeds, D.D. Zhang, T., and Weiss, G.M. 2021. Mining course groupings using academic performance. In *Proceedings of The 14th International Conference on Educational Data Mining*, International Educational Data Mining Society, Paris France, June 29-July 2, 804-808.
- [5] Lilienfeld, E. 2016. How student evaluations are skewed against women and minority professors. *The Century Foundation*. <https://tcf.org/content/commentary/student-evaluations-skewed-women-minority-professors>.
- [6] Riad-Zaky, M., Weiss, G.M., and Leeds, D.D. 2022. Course Grade Analytics with Networks (CGAN) [computer software], available <https://www.cis.fordham.edu/edmlab/software>.
- [7] Uttl, B., White, A. C., and Gonzalez, D. W. 2017. Meta-analysis of faculty’s teaching effectiveness: Student Evaluation of teaching ratings and student learning are not related. *Studies in Educational Evaluation.*, 54, 22-42. DOI=<https://doi.org/10.1016/j.stueduc.2016.08.007>.
- [8] Vliieger, P., Jacob, B., Stange, K. 2016. Measuring instructor effectiveness in higher education. *National Bureau of Economic Research*. <https://www.nber.org/papers/w22998>.
- [9] Wachtel, H.K. 1998. Student evaluation of college teaching effectiveness: a brief review, *Assessment & Evaluation in Higher Education*, 23:2, 191-212. DOI=<https://doi.org/10.1080/0260293980230207>.

APPENDIX

This appendix provides additional information related to the two thresholds that were discussed in Section 4. The relevant underlying data distributions are shown, which inform the choice of specific threshold values.

Figure 3 shows the distribution of the standard deviation values for grades at the course section level. There are quite a few sections

with grade standard deviation near zero, most likely due to small project-based courses, where instructors often assign grades of “A”. As discussed in Section 3.1, sections with grade standard deviations below *MinSD* are removed since student performance cannot be effectively measured in such cases. Threshold values of 0.1, 0.2, 0.3, 0.4, and 0.5, were evaluated before a *MinSD* value of 0.2 was selected; that value was selected because it ensures a reasonable level of variance in the course grades while retaining most of the course sections.

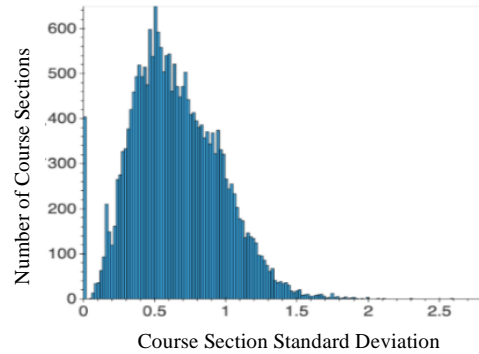


Figure 3. Distribution of Section Grade Standard Deviations

Figure 4 shows how the number of course pairs vary, in log scale, based on the number of students in the course pair. This number is based on the students in the first course in the pair taught by a particular instructor. The figure is used to help select the *MinStudents* threshold defined in Section 3.5, which removes course pairs with too few students. Most pairings have less than 100 students, even though course pairs are aggregated over all relevant course sections; this occurs because many course pairs involve disparate courses in different disciplines. Our results are based on *MinStudents*=50. A larger threshold would increase the reliability of our instructor benefit scores but eliminate too many course pairs.

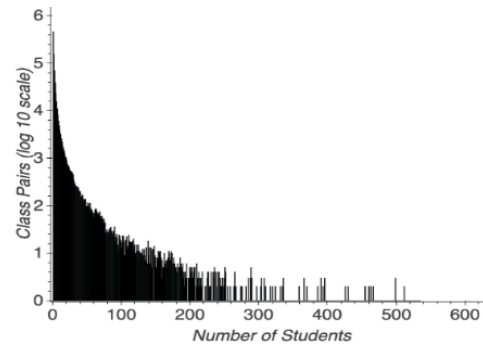


Figure 4. Number of Students in Course Pairs by Instructor

Modeling study duration considering course enrollments and student diversity

Niels Seidel

FernUniversität in Hagen, Research Cluster D²L², Universitätsstr. 11, 58084 Hagen, Germany
niels.seidel@fernuni-hagen.de

ABSTRACT

Students are self-determined to choose degree programs and courses at their own pace. However, this variety of choices can lead to a long duration of study, especially in part-time distance learning. Hence, this paper aims to explore data on course enrollments of students pursuing bachelor's and master's degrees in Computer Science and Mathematics at a European distance-based German university to uncover predictors for study duration. Distance students have highly diverse backgrounds, which might also be represented in their enrollment behavior and duration of study. Thus, it is vital to analyze this behavior to identify bottlenecks and adjust instructions. We employed a Multiple Regression Analysis with a Genetic Algorithm for model selection to uncover predictors that lengthen or shorten the study duration. For model selection, we considered demographic data, modes of study, enrollment behaviors, and individual courses. We used the method to find predictors within the data of 1898 students who graduated in at least one of the five study programs offered by the Faculty of Mathematics and Computer Science between 1999 and 2019. The enrollment behavior strongly predicts the duration of study compared to demographic and study-behavior predictors. Individual courses are good predictors for specific study programs.

Keywords

course enrollment, study duration, multiple linear regression

1. INTRODUCTION

In accordance with the Humboldtian model of higher education, students are self-determined to choose degree programs and courses at their own pace. However, this variety of choices can lead to a long duration of study, especially in part-time distance learning. At many universities, a long duration of study was not seen as a problem as long as sufficient capacity was available. In the OECD countries, the proportion of young people between 25 and 34 years of age with a university degree rose by a total of 14 percentage points

between 2000 and 2013 [27]. In Germany for instance, the number of students increased by 1,146,262 (63%) between 1998/99 and 2021/22¹. At the same time, the capacities for teaching did not increase at the same level. In this paper, we take a look at the duration of study as a driver for high student number.

Study programs are designed so that students graduate within a defined amount of time. If students exceed this regular period of study, it has consequences for the students, the teachers, and the respective faculty. Additional semesters cost a student time and effort to repeat courses and exams. This can also be accompanied by financial costs for tuition fees and a late entry into professional life or a higher career level. In addition, there are psychological burdens. For instructors, longer study durations mean an increase in the amount of supervision required due to the need to retake courses and exams. This is apparent in the supervision ratio which is defined as the number of students per teacher. From a faculty perspective, long-term students need to be considered for capacity planning. Just like the number of new program enrollments, the number of graduations is part of the target agreements or key performance indicators considered by the university management and ultimately the federal or state ministries of education. Today, higher educational organizations are placed in a very highly competitive environment. The analysis, presentation, and data mining is one approach to tackle challenges in the organization of study programs.

The causes of protracted studies are not necessarily due to a lack of motivation, performance, or effort on the part of students. Behavioral factors in the choice of one or more courses of study, as well as the distribution of the workload over the semesters, can have a major influence on the time to degree. Past studies have shown that differences in enrollment behavior are related to student diversity factors [2]. The manifestations of these factors vary by country/culture, university type, institution, and program respective subject domain. Further factors for a long study duration are under the influence of teachers. Repetition of courses and exams can be an indicator of high difficulty, but also of inadequate instructional design or exams with low pass rates. Thus, it is vital to analyze enrollment behavior to identify bottlenecks and adjust instructions. Other reasons for a slowed down study progress result from organizational bottlenecks, such

N. Seidel. Modeling study duration considering course enrollments and student diversity. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 621–628, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852956>

¹See <https://www.datenportal.bmbf.de/portal/de/K254.html> (accessed 2022/05/08).

as overfilled courses, missing or too late reexaminations, annual instead of semesterly course offers and examinations.

Hence, this paper aims to explore data on course enrollments of students pursuing bachelor's and master's degrees in Computer Science and Mathematics at a German distance-based university to uncover predictors for study duration. Our study aims at providing initial insights into enrollment processes of German distance learning students. In particular, we are going to focus on one research question (RQ): *(RQ1) What predictors significantly influence the duration of study?* To answer this question we employed a Multiple Linear Regression Analysis with a Genetic Algorithm for model selection to uncover predictors that lengthen or shorten the study duration. For model selection, we considered demographic data, modes of study, enrollment behaviors, and individual courses. We used the method to find predictors within the data of 1898 students who graduated in at least one of the five study programs offered by the Faculty of Mathematics and Computer Science between 1999 and 2019.

Identifying predictors associated with time to graduation can help educators design better degree plans, and students make informed decisions about future enrollments. Distance students have highly diverse backgrounds, which might also be represented in their enrollment behavior and duration of study. Thus, it is vital to analyze this behavior to identify bottlenecks and adjust instructions.

2. RELATED WORKS

There exist various studies focusing on enrollment data. In this section, we provide an overview of the background and intent for the analysis of these data and shed a light on the data and methods used.

Most of the research on enrollment data relates to educational institutions in the Anglo-American world. Among the cited literature in this paper, only two papers refer to African [34, 13] and three to European institutions of higher education [7, 33, 3]. The majority of the works come from traditional universities compared to distance learning universities as referred by [33] or MOOCs [31].

The intentions for analyzing enrollment data range from descriptive analysis, prediction to the preparation of interventions. [33] identify factors contributing to students continuing for the duration of their distance learning studies and completing their degree. The motivation for enrollment to computer science degree programs has been explored by Duncan et al. [15]. Age, gender, and demographic trends in motivation (goals, opportunities, and assurance of goal achievement) for enrollment have been analyzed and significant motivation differences regarding gender and age have been reported. Sahami et al. [30] explored the phenomena of performance decline using the computer science enrollments data from Stanford University and found that despite increased enrollments, student performance remains stable. Analysis is conducted on different scales such as courses [7], study programs [35, 34] and faculties [35, 34]. [24] and [9] for instance, analyzed changes in the enrollment and study progress before and after policy changes. [35] focuses on students' experiences of guidance in relation to their study progress and perceptions of their learning outcomes. The impact of co-enrollment was studied by [8] and [37]. The prediction of dropout (e.g. [10, 22, 7]), study performance

(e.g. [11, 14, 17, 25, 29, 38, 39]), and future enrollments (e.g. [20, 23, 36]) gained a lot of attention in the last years. Prediction of time to degree were employed by [18] and [21]. [6] identify potential predictors of academic success including the time to graduation for Ph.D. students. Age, sex, employment institution, mentor experience, and tuition subsidy had no influence on the time to graduation and completion rate. [35] predicted slow study progress from self-report data using Binary Logistic Regression. [24] identified factors affecting time to bachelor's degree attainment. Dahdouh et al. used association rules mining over course enrollments for recommendations of further study paths [12]. The rules are used for recommending suitable courses to students based on their behavior and preferences. [7] investigated bottlenecks of learning progress in order to support the student advisory services, while [28] make use of enrollment data to prepare re-enrollment campaigns.

Data collected from university information systems has been proved to be the source of helpful information (e.g. [24, 9, 4]) for improving study processes and educational decisions. However, due to a strong data protection culture, some European universities tend to interpret the European data protection regulations (GDPR) very strictly. Even within institutions researchers do not get access to personal data and are also not allowed to link anonymized data. Student performance data such as grades are considered particularly sensitive. Another common source for investigating enrollment behavior comes various forms of self-reports including surveys (e.g. [26, 29, 13, 13, 31, 19]). The used variables cover a broad range that reflects cultural and institutional conditions. For example, the housing situation was studied in countries where campus universities are found [5].

Subgroups including their intersections have been rarely considered [35, 3, 2]. [3] for instance, identified differences in study success and early dropout between minority and majority students in economics which can be attributed to differences in high school education, but not on academic and social integration. [2] considered dimensions underpinning students' study philosophy towards teaching, learning, and study for different groupings and subgroup interactions (e.g. age, sex, ethnicity, study discipline, academic performance). The definition of student profiles [7, 13] is an approach coming from social science which can be helpful to distinguish and explain patterns of subgroups.

The analytical methods used for enrollment analyses include frequent item mining [1, 12], sequence mining [1, 9], Clustering [34], Social Network Analysis [37], Latent Profile Analysis [13], and Linear/Logistic Regression [24, 35, 34, 6]. For example, Elbadrawy et al. [16] used sequence mining via the so-called Universal discriminating Pattern Mining framework capable of mining enrollment patterns from groups of low and high-performing students to enable educators for better degree planning. [26] applied an investment theory to predict the degree of commitment. The application of a Multiple Linear Regression by [5] and [24] underlines its advantages with regard to traceability, explainability, and the possibility of deriving interventions.

3. METHODS

3.1 Data

The data set contains 1489 bachelor students and 1014 master's students who enrolled in 1999 to 2016 and finished the degree until 2019. The collected data include student en-

rollments to courses during their studies, information about completion of the degree, and a list of courses required to complete the degree. In addition, the enrollment data do not contain information on whether a student finished a course successfully since different departments carry out the oral and written examinations at the Faculty. University data protection rules restrict the use and analysis of the exam results. By enrolling in the program, students gave their consent to the processing of the data used in this analysis. To further ensure data privacy the unique identifiers of the students have been pseudonymized in order to prevent linking with other datasets and to prohibit the identification of individual students. However, the identification of individuals cannot be ruled out, the data set will only be provided on request instead of being published.

The available data includes demographic data as well as information on the enrolled programs and courses. From this information, four diversity dimensions will be categorized with regard to (i) demographics, (ii) study behavior, (iii) enrollment behavior, and (iv) course impact. While the first three categories are related to the students, the latter refers to organizational and didactical aspects mainly influenced by the responsible teachers.

The demographic data available contains the age at program admission, gender, and the completion of previous bachelor's or master's degrees. The age ranges between 14 and 69 in all study programs. Detailed demographic information per program are listed in Tab. 1.

From the program enrollment data, we derive study behavior information. Students at the Faculty of Mathematics and Computer Science can enroll in up to three programs at the same time. These programs are prioritized by the student (cf. Program priority). For each program, students can decide whether to study full-time or part-time which has an effect on the expected study duration. The duration of study in part-time study is half as long as the full-time duration. Furthermore, Master's programs distinguish consecutive study after completing a related bachelor's degree and non-consecutive study. A second degree is stated if a student already achieved a degree on the same level (e.g. a second bachelor's degree). Listener status describes the opportunity to join a program as a guest or listener without the obligation to achieve a degree.

The enrollment behavior is described by the number and variety of course enrollments per semester and in total. For the first three semesters first-time and re-enrollments are counted separately (e.g. *Enrollments 1st semester*, *Repetitions 2nd semester*). For the number of unique courses, we distinguish between courses offered at the Faculty (*Different Faculty courses*) and those offered at another faculty (*Different other courses*). Semesters without any enrollments are described as *semesters off*.

Furthermore, 25% of the most frequently enrolled course have been dummy-coded for each student representing the fourth diversity dimension.

3.2 Multiple Linear Regression

For each study program, the student data was represented in a Learner Profile including the before mentioned data about the demographics, study behavior, and enrollment behavior as well the binary information about the most frequently enrolled courses.

Outliers regarding the total number of different enrolled courses, the total course repetitions, and the repeating enrollments have been removed. Values above the mean plus three times the standard deviation have been considered as outliers. Finally, 884 B.Sc. and 1014 M.Sc students remained in the dataset.

The mentioned variables have been selected from the Learner Profile and used to produce model formulas. These formulas are passed to a fitting function. The variables in the formula correspond to the data in the Learner Profile. The duration of study was defined as the dependent variable. The remaining variables were used as independent variables in the formulas. By default, an intercept is included in all models. Due to the initial use of a large number of variables, it is necessary to find a simpler model based on fewer variables. Instead of trying all candidates for a suitable model with an unapplicable brute force approach, the candidate set is explored by a Genetic Algorithm (GA). A GA can readily find the best models without fitting all possible models. For the GA the formula is encoded as a sequence of binary values. This sequence forms a population that will undergo an evolution by adapting certain bits to form a new generation. The genetic algorithm keeps track of a population of models and their size. Asexual reproduction, sexual reproduction from parental generations, and immigration are the three methods used to create the next generation of models.

As decision criterion the Akaike Information Criterion (AIC) is used. It is defined by

$$AIC = -2l(\hat{\beta}_M, \sigma^2) + 2|M + 1| \quad (1)$$

with $l(\hat{\beta}_M, \sigma^2)$ as the maximum value of the log-likelihood and M as the number of variables present in the current model. On the one hand, one can see that the AIC value is negatively directed, which is why the goal of model selection is to minimize this value. On the other hand, a high number of variables is penalized. Thus, a too complex model is prevented. The models are fitted to every generation by using the AIC values to calculate each model's fitness, w . The i th model's fitness is calculated as follows:

$$w_i = \exp(-(AIC_i - AIC_{best})) \quad (2)$$

where AIC_{best} is the best AIC in the current population of models. Lower AIC means higher fitness. Inference was aided by point and interval (95% CI) estimates, the goodness of fit measures, AIC, and p values.

In order to measure and compare the goodness of a fitted model we compute the Cragg-Uhler *Pseudo-R*². *Pseudo-R*² is defined as one minus the ratio of the residual deviance and the intercept (null deviance):

$$R^2 = 1 - \frac{ResidualDeviance}{NullDeviance} \quad (3)$$

R^2 describes the deviation of the current model between 0 and 1, whereas 0 means total deviation and 1 a complete congruence.

4. RESULTS AND DISCUSSION

Appendix A.1 provides an overview of the fitted models and the number of predictors with regard to the four categories of diversity dimensions. Except for the B.Sc. CS the values for R indicate a good model fit. For the smaller number of graduates in the M.Sc. in Mathematics a very good was

Table 1: Demographic information about the students who graduated in the Computer Science (CS) and Mathematics programs

Program name	B.Sc. CS	M.Sc. Practical CS	M.Sc. CS	B.Sc. Mathematics	M.Sc. Mathematics
Time range	1999-2019	2003-2019	2003-2019	2000-2018	2003-2018
N (mean \pm sd)					
Women	454	16	120	130	9
Men	634	153	690	180	33
Total	686	169	803	198	42
Age at admission (years mean \pm sd)					
Women	31.77 \pm 6.14	31.56 \pm 5.39	32.53 \pm 7.84	28.39 \pm 7.88	31.78 \pm 10.44
Men	30.44 \pm 6.24	29.86 \pm 5.42	31.61 \pm 6.82	30.79 \pm 7.88	30.45 \pm 7.93
Total	30.69 \pm 6.23	30.02 \pm 5.42	31.74 \pm 6.98	30.29 \pm 7.89	30.74 \pm 8.41
Time to degree (semesters mean \pm sd)					
Women	14.15 \pm 6.19	10.19 \pm 4.81	5.58 \pm 3.31	11.06 \pm 3.7	5.89 \pm 1.05
Men	11.89 \pm 6.18	8.5 \pm 3.82	5.89 \pm 3.56	10.22 \pm 4.83	7.91 \pm 3.74
Total	12.31 \pm 6.24	8.66 \pm 3.94	5.85 \pm 3.52	10.4 \pm 4.61	7.48 \pm 3.44

achieved. The AIC and BIC measures are not suitable to make comparisons between the programs but relate to the model complexity. The model of the bachelor of CS appears to be the most complex with 29 predictors. Here again, the M.Sc. in Mathematics stands out with simpler model of 8 predictors.

The four diversity dimensions have a different influence on the models. In general, it can be said that demographic factors and study behavior predicting the study duration less than the enrollment behavior. The effect of individual courses depend on the study program.

The fitted linear regression models for predicting variables influencing the duration of study in each of the five study programs are presented in the Appendix A.2. The size of the coefficients expresses the number of semesters by which the study is extended or, if negative, shortened. For example, if a student takes 3 courses in the Bachelor CS in the 3rd semester, the duration of study is shortened by 3 times -0.38, i.e. by 1.14 semesters. Binary represented values like gender or taking a certain course correspond to factor 1. As stated before, age has no significant impact on study duration. Note, that the coefficient for the age is multiplied by the number of years. As a result, this apparently small coefficient may predict the study duration of elderly students. Also the gender impact is comparatively small, but recognizable with opposite direction in the CS bachelors and Practical CS masters' programs. For the same two programs the existence of a past degree predicts the time to degree. While the length of study for students in the Bachelor CS is shortened by the experience gained in another program, the length of study is lengthened for students in the Master Practical CS. Studying in multiple programs at the same time can be beneficial for the overall study duration. This can be explained by the fact that examination credits from one study program can be credited in the thematically related study programs of the faculty. Thus, a successfully completed examination can be used in several study programs. However, for the M.Sc. CS additional activities on other programs is at the expense of the duration of study. The enrollment to courses of other faculties extends the time needed for completion.

As expected, the total course repetition and the variety of chosen Mathematics-related or CS-related course strongly predict study duration. A single semester off lengthens the study duration by more than one semester.

5. SUMMARY AND OUTLOOK

In this paper, we explored data on course enrollments of students pursuing bachelor's and master's degrees in Computer Science and Mathematics at a European distance-based German university to uncover predictors for study duration. We tried to consider the highly diverse backgrounds of distance-learning students that are represented in a restricted and pseudonymized dataset consisting only of information on current and past study programs and enrolled courses. From this information, Learner Profiles have been created. These profiles contained measures that are potentially suitable for describing influencing factors for the duration of study. Instead of predicting the time of study completion for future cohorts, we used them to describe and analyze the past student (and teacher) behavior. We find it is vital to analyze this behavior to identify bottlenecks and adjust instructions as well as the organization of study programs.

We employed a Multiple Regression Analysis with a Genetic Algorithm for model selection to uncover predictors that lengthen or shorten the study duration. For the models, we considered demographic data, study behavior, enrollment behaviors, and individual courses. We used the method to find predictors within the data of 1898 students who graduated in at least one of the five study programs offered by the Faculty of Mathematics and Computer Science between 1999 and 2019. With regard to RQ1 the enrollment behavior strongly predicts the duration of study compared to demographic and study-behavior predictors. Individual courses are good predictors for specific study programs.

As a next step, we want to identify changes in the fitted models over time. The considered time range of almost 20 years included many changes of regulations, tuition fees, and teaching staff. Similar to the work of [24] and [9] we want to trace predictors over time in order to recognize relevant trends for teachers and faculty managers. With this regard, we also would like to continue our past research about student course recommenders [32].

Acknowledgments

This research was supported by the Research Cluster "Digitalization, Diversity and Lifelong Learning – Consequences for Higher Education" (D²L²) of the FernUniversität in Hagen, Germany.

References

- [1] Z. Abdullah, T. Herawan, N. Ahmad, and M. M. Deris. Extracting highly positive association rules from students' enrollment data. *Procedia-Social and Behavioral Sciences*, 28:107–111, 2011.
- [2] M. Alauddin and A. Ashman. The changing academic environment and diversity in students' study philosophy, beliefs and attitudes in higher education. *Higher Education Research & Development*, 33(5):857–870, 2014.
- [3] I. J. M. Arnold. Ethnic minority dropout in economics. *Journal of Further and Higher Education*, 37(3):297–320, 2013.
- [4] K. E. Arnold and M. D. Pistilli. Course Signals at Purdue: Using Learning Analytics to Increase Student Success. *Proceedings of the 2nd LAK Conference*, pages 267–270, 2012.
- [5] S. Beekhoven, U. D. Jong, and H. V. Hout. The impact of first-year students' living situation on the integration process and study progress. *Educational Studies*, 30(3):277–290, 2004.
- [6] B. Benzon, K. Vukojevic, N. Filipovic, S. Tomić, and M. G. Durdov. Factors that determine completion rates of biomedical students in a PhD programme. *Education Sciences*, 10(11):1–8, 2020.
- [7] A. Böttcher, V. Thurner, T. Häfner, and S. Ottinger. Adaptierung von Beratungsangeboten auf der Basis von Erkenntnissen aus der Analyse von Studienverlaufsdaten. In *9. Fachtagung Hochschuldidaktik Informatik (HDI)*, pages 57–64, 2021.
- [8] M. G. Brown, R. Matthew DeMonbrun, and S. D. Teasley. Conceptualizing Co-enrollment: Accounting for student experiences across the curriculum. In *LAK '18 8th International Conference on Learning Analytics and Knowledge, March 7-9, 2016*, pages 305–309, Sydney, 2018. ACM.
- [9] D. Canales Sánchez, T. Bautista Godínez, J. G. Moreno Salinas, M. García-Minjares, and M. Sánchez-Mendiola. Academic trajectories analysis with a life-course approach: A case study in medical students. *Cogent Education*, 9(1):2018118, dec 2022.
- [10] H. E. Caselli Gismondi and L. V. Urrelo Huiman. Multilayer Neural Networks for Predicting Academic Dropout at the National University of Santa - Peru. In *2021 International Symposium on Accreditation of Engineering and Computing Education (ICACIT)*, pages 1–4, 2021.
- [11] R. Costa-Mendes, T. Oliveira, M. Castelli, and F. Cruz-Jesus. A machine learning approximation of the 2015 Portuguese high school student grades: A hybrid approach. *Education and Information Technologies*, 26(2):1527–1547, 2021.
- [12] K. Dahdouh, A. Dakkak, L. Oughdir, and A. Ibriz. Large-scale e-learning recommender system based on Spark and Hadoop. *Journal of Big Data*, 6(1), 2019.
- [13] M. De Clercq, B. Galand, and M. Frenay. One goal, different pathways: Capturing diversity in processes leading to first-year students' achievement. *Learning and Individual Differences*, 81:101908, 2020.
- [14] E. Demeter, M. Dorodchi, E. Al-Hossami, A. Benedict, L. Slattery Walker, and J. Smail. Predicting first-time-in-college students' degree completion outcomes. *Higher Education*, 2022.
- [15] A. Duncan, B. Eicher, and D. A. Joyner. Enrollment motivations in an online graduate cs program: Trends and gender- and age-based differences. In *Annual Conference on ITiCSE*, pages 1241–1247, 2020.
- [16] A. Elbadrawy and G. Karypis. UPM: Discovering Course Enrollment Sequences Associated with Success. In *Proceedings of the 9th LAK Conference*, LAK19, pages 373–382, New York, NY, USA, 2019. Association for Computing Machinery.
- [17] A. Gambini, M. Desimoni, and F. Ferretti. Predictive tools for university performance: an explorative study. *International Journal of Mathematical Education in Science and Technology*, pages 1–27, jan 2022.
- [18] T. Hailikari, R. Sund, A. Haarala-Muhonen, and S. Lindblom-Ylänne. Using individual study profiles of first-year students in two different disciplines to predict graduation time. *Studies in Higher Education*, 45(12):2604–2618, 2020.
- [19] T. Hailikari, T. Tuononen, and A. Parpala. Students' experiences of the factors affecting their study progress: differences in study profiles. *Journal of Further and Higher Education*, 42(1):1–12, 2018.
- [20] N. A. Haris, M. Abdullah, N. Hasim, and F. Abdul Rahman. A study on students enrollment prediction using data mining. In *ACM IMCOM 2016: Proceedings of the 10th International Conference on Ubiquitous Information Management and Communication*, 2016.
- [21] S. Herzog. Estimating student retention and degree-completion time: Decision trees and neural networks vis-à-vis regression. *New Directions for Institutional Research*, 2006(131):17–33, 2006.
- [22] B. Jeon and N. Park. Dropout Prediction over Weeks in MOOCs by Learning Representations of Clicks and Videos. *CoRR*, abs/2002.0, 2020.
- [23] M. S. Kiran, E. Siramkaya, E. Esme, and M. N. Senkaya. Prediction of the number of students taking make-up examinations using artificial neural networks. *International Journal of Machine Learning and Cybernetics*, 13(1):71–81, 2022.
- [24] W. E. Knight. Time to Bachelor's Degree Attainment: An Application of Descriptive, Bivariate, and Multiple Regression Techniques. *IR Applications*, Volume 2, September 8, 2004. 2004.
- [25] M. F. Musso, C. F. R. Hernández, and E. C. Cascallar. Predicting key educational outcomes in academic trajectories: a machine-learning approach. *Higher Education*, 80(5):875–894, 2020.

- [26] S. Noxel and L. Katunich. Navigating for Four Years to the Baccalaureate Degree. AIR 1998 Annual Forum Paper. Technical report, Ohio State University, 1998.
- [27] OECD. *Education at a Glance 2020*. 2020.
- [28] J. C. Ortagus, M. Tanner, and I. McFarlin. Can Re-Enrollment Campaigns Help Dropouts Return to College? Evidence From Florida Community Colleges. *Educational Evaluation and Policy Analysis*, 43(1):154–171, 2021.
- [29] H. Prabowo, A. A. Hidayat, T. W. Cenggoro, R. Rahutomo, K. Purwandari, and B. Pardamean. Aggregating Time Series and Tabular Data in Deep Learning Model for University Students’ GPA Prediction. *IEEE Access*, 9:87370–87377, 2021.
- [30] M. Sahami and C. Piech. As CS Enrollments Grow, Are We Attracting Weaker Students? pages 54–59, 2016.
- [31] E. Schneider and R. F. Kizilcec. ”Why Did You Enroll in This Course?”: Developing a Standardized Survey Question for Reasons to Enroll. In *Proceedings of the First ACM Conference on Learning Scale Conference, L@S ’14*, pages 147–148, New York, NY, USA, 2014. Association for Computing Machinery.
- [32] N. Seidel, M. C. Rieger, and A. Walle. Semantic Textual Similarity of Course Materials at a Distance-Learning University. In T. W. P. And, P. B. And, S. I. H. And, K. K. And, and Y. Shi, editors, *Proceedings of 4th CSEDM Workshop co-located with the EDM 2020 Conference*. CEUR-WS.org, 2020.
- [33] J. Simons, S. Leverett, and K. Beaumont. Success of distance learning graduates and the role of intrinsic motivation. *Open Learning*, 35(3):277–293, 2020.
- [34] F. Siraj and M. A. Abdoulha. Uncovering hidden information within university’s student enrollment data using data mining. In *2009 Third Asia International Conference on Modelling & Simulation*, pages 413–418. IEEE, 2009.
- [35] T. Skaniakos, S. Honkimäki, E. Kallio, K. Nissinen, and P. Tynjälä. Study guidance experiences, study progress, and perceived learning outcomes of Finnish university students. *European Journal of Higher Education*, 9(2):203–218, 2019.
- [36] J. Ward. Forecasting enrollment to achieve institutional goals. *College and University*, 83(3):41, 2007.
- [37] K. A. Weeden, B. Cornwell, and B. Park. Still a Small World? University Course Enrollment Networks before and during the COVID-19 Pandemic. *Sociological Science*, 8:73–82, 2021.
- [38] S. K. Yadav and S. Pal. Data mining: A prediction for performance improvement of engineering students using classification. *arXiv preprint arXiv:1203.3832*, 2012.
- [39] M. Yağcı. Educational data mining: prediction of students’ academic performance using machine learning algorithms. *Smart Learning Environments*, 9(1):11, 2022.

APPENDIX

A. LINEAR REGRESSION MODELS

A.1 Overview of Linear Regression Models

Table 2: Overview of the fitted Linear Regression Models of students' time to degree

Program	B.Sc. CS	M.Sc. CS	M.Sc. Practical CS	B.Sc. Mathe	M.Sc. Mathe
Model goodness					
R ²	0.67	0.81	0.88	0.82	0.92
AIC	1380.39	611.59	2306.13	324.20	115.11
BIC	1433.69	656.85	2388.48	353.61	129.11
Number of predictors					
Demographic-related	3	0	3	0	0
Study-related	3	2	1	2	2
Enrollment-related	11	10	11	5	6
Course-related	13	2	4	8	1
Total	29	13	17	14	8

* p<.1, ** p<.01, *** p<.001

A.2 Coefficients of the best fit Linear Regression Models

Table 3: Coefficients of the best fit Linear Regression Models of students' time to degree

Coefficient	B.Sc. CS	M.Sc. CS	M.Sc. Pract. CS	B.Sc. Mathe	M.Sc. Mathe
(Intercept)	10.90***	4.11**	1.93***	9.49***	3.27***
Age	0.04	-	0.01*	-	-
Male	-1.02*	-	0.19	-	-
previousDegreesMaster	-1.61*	-	0.81*	-	-
Fulltime study	-1.06*	-	-0.25*	-	-
Programme priority	-0.79	1.67	-	-3.04*	3.27***
Second degree	0.68	1.28*	-	1.73**	-
Semesters off	0.95***	1.44*	1.13***	1.48***	3.27***
Total course repetitions	0.23***	0.49***	0.46***	0.30***	0.49***
Different CS courses	-0.05	0.2***	0.18***	0.10**	-
Different other courses	0.04	-	0.26***	-	0.20***
Enrollemnts 1st semester	-0.17*	-0.19*	-0.61***	-	-0.23*
Enrollments 2nd semester	-0.05	-0.28*	-0.46***	-	-
Enrollments 3rd semester	-0.38**	-0.38***	-0.34***	-	-
Repetitions 1st semester	-0.37	-0.89***	-0.71***	-1.00**	-
Repetitions 2nd semester	-0.35	-0.75***	-0.57***	-	-1.03***
Repetitions 3rd semester	-0.12	-0.8**	-0.17*	-1.72***	-0.62*
Course 1144	-	-	-	-5.22***	-
Course 1145	-	-	-	4.64***	-
Course 1202	-	-	-	-1.45*	-
Course 1358	-	-	-	-	-1.81*
Course 1359	-	-	-	-	2.38*
Course 1361	-	-	-	-1.54*	-
Course 1584	1.49**	-	-	-	-
Course 1613	-0.07	-	-	-	-
Course 1618	0.49	-	-	-	-
Course 1657	-1.43	-	-	-	-
Course 1658	0.34	-	-	-	-
Course 1661	0.63	-	-	-	-
Course 1666	-	-	0.27**	-	-
Course 1671	0.14	-	-	-	-
Course 1678	-0.46	-	-	-	-
Course 1793	0.21	-	-	-	-
Course 1801	0.06	-	-	-	-
Course 1814	-	-	0.33***	-	-
Course 1853	-	0.54*	-	-	-
Course 1866	0	-	-	-	-
Course 1895	0.01	-	-	-	-
Course 1896	1.21*	-	-	-	-

* p<.1, ** p<.01, *** p<.001

Generalized Sequential Pattern Mining of Undergraduate Courses

Daniel D. Leeds, Cody Chen, Yijun Zhao, Fiza Metla, James Guest, and Gary M. Weiss
Computer and Information Science Department
Fordham University, New York, NY
{dleeds, cchen187, yzhao11, fmetla, jguest2, gaweiss}@fordham.edu

ABSTRACT

University students have a great deal of freedom in deciding the order in which to take their courses. In this paper we apply the Apriori-based Generalized Sequential Pattern (GSP) algorithm to undergraduate course data from a large university in order to identify frequent course sequences. Course sequencing results are primarily generated at the department level, with a special focus on Computer Science courses. This paper also introduces the course sequence flow diagram, which compactly represents a large amount of course sequencing information in an intuitive visual form. Our results and associated flow diagrams can help to answer a variety of important questions, such as: what course sequences are most common, how are courses between different departments ordered, and when are courses taken in an order that may contradict the advice given by academic advisors? In this paper we show that this form of descriptive data mining can identify standard core curriculum and pre-health sequences of study, as well as computer science courses that are either artificially pushed to the end of a student's program of study or taken earlier than would be recommended.

Keywords

Sequence mining, association analysis, course sequencing, educational data mining.

1. INTRODUCTION

The order in which university students take their courses is only partially constrained by course prerequisites and university policies. However, course sequencing is important since it impacts student learning and can impact student grades. This was demonstrated by one of our research group's recent studies that looked at pairs of courses taken in both possible orderings and showed that the different orderings produced different grade performance [4]. Other work has looked at the sequences that courses in specific disciplines, such as communications [7] and psychology [2], are taken in, and assessed how these impact student learning. Some work looks more generally at course selection and how it impacts student grades [5] or time to graduation [6].

Prior work focuses mainly on assessing the impact of specific course sequences rather than on identifying or characterizing course sequences. This paper focuses on the descriptive data mining task of identifying common course sequences and how to best represent this information. These sequences are of intrinsic value, providing insight into how our curricula operate in practice. They

D. Leeds, C. Chen, Y. Zhao, F. Metla, J. Guest, and G. Weiss. Generalized sequential pattern mining of undergraduate courses. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 629–633, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852978>

can identify course prerequisite structures and interrelationships between departments. They can also expose courses taken in an unexpected, and perhaps inadvisable, order. These insights can be used to modify and improve academic advising and inform curricular changes, such as by modifying course prerequisites.

In this paper we use the Apriori-based Generalized Sequential Pattern (GSP) algorithm [9] to identify frequent k -sequences, where a k -sequence contains k courses taken in a specific sequential order. We apply this method to courses at the department level and in some cases across departments. We show how these frequent k -sequences can be used to form a course sequence flow diagram that encapsulates the knowledge of many sequential patterns in a simple and useful visual form. We then show how the flow diagrams and associated results can be used to gain useful academic insights, which can inform academic advising and perhaps even changes to the curricula (e.g., modifications to prerequisites or course numbering). Our most detailed analysis focuses on the Computer Science department, building off our familiarity with our home department.

In Section 2 we describe the Apriori and GSP algorithms. Section 3 describes the course-enrollment data set provided by our university and the steps necessary to transform this data into the student-level course sequences suitable for mining by the GSP algorithm. In Section 4 we present our results in the form of frequent k -sequences, introduce the course sequence flow diagram, and extract insights from these results by utilizing our curricular domain knowledge. Section 5 provides our broader conclusions, study limitations, and plans for future work.

2. SEQUENCE MINING

This paper extracts common course sequences from undergraduate course enrollment data. It relies on the generalized sequential pattern mining framework [9], which is an extension to the Apriori association rule mining algorithm that takes ordering into account. Whereas Apriori generates a set of frequent itemsets, GSP generates a set of frequent sequences, and, in the context of this study, a set of frequent course sequences. Minimum support is used analogously to its use in Apriori, so a specific sequence is considered frequent if it occurs in more than *minsup* sequences. We briefly introduce the well-known Apriori algorithm and then describe the GSP algorithm as an extension to Apriori.

The Apriori association rule mining algorithm [1] was developed to provide retailers with information about what items are usually purchased in the same transaction. Formally, let $I = \{I_1, I_2, I_3, \dots, I_d\}$ be the distinct items and let $T = \{T_1, T_2, T_3, \dots, T_n\}$ be the set of customer transactions, where each T_i contains a subset of items in I . A k -itemset X is a collection of exactly k items:

$$X = \{X_1, X_2, X_3, \dots, X_k\}, \text{ where } X_i \in I \ \forall i, \text{ and } |X| < d.$$

Given a set of transactions T , the *support* of a k -itemset is the fraction of the transactions T that contain all of the k items in the k -itemset (e.g., if $k=2$ and the items are *milk* and *cereal*, then

support is the fraction of transactions that contain *milk* and *cereal*). Association rule algorithms like Apriori [1] and FP-growth [3] identify all *frequent k*-itemsets. These algorithms also include a second step that generates association rules from the itemsets, but our study does not need to generate rules, only frequent sequences, so we focus only on the first step.

The GSP algorithm [9] is a modest extension of the Apriori algorithm. The extension involves making the Apriori algorithm sensitive to the order of items (i.e., courses) in each transaction. There are two extensions that need to be made to the Apriori algorithm. The first is to extend the candidate generation phase to generate ordered itemsets and the second is to account for item ordering when computing the support for an itemset (i.e., sequence).

3. METHODOLOGY

This section describes our initial data set, how we transform this data into a form suitable for the GSP algorithm, and the impact of the *minsup* threshold on the resulting course sequences.

3.1 Initial Course-Enrollment Data Set

This study is based on eight years of undergraduate course enrollment data from Fordham University. Each entry in the initial data set represents the enrollment of a specific student in a course section, along with their grade in the course and the semester and year that they took the course. To preserve privacy the student identifiers were mapped to new, but consistent, values and course sections with fewer than five students were dropped. The data set contains 473,527 records that collectively cover 24,969 distinct students. For the purpose of this study all sections of the same course are considered equivalent. Even with these privacy protections we cannot share the course-enrollment data, due to strict privacy laws.

3.2 Student Course-Sequence Data Set

The initial data is not in the format expected by the GSP algorithm since it expects each entry to include an ordered sequence of courses. Thus we transform the initial data set into a student course-sequence data set by separating the courses for each student and then ordering these courses chronologically. Courses that are taken during the same semester are ordered lexicographically using the course identifier. Below is a sample course sequence record, which starts with the student-ID and is followed by five courses:

S245632: CS1000, CS2000, HIST1200, CS3462, CS2250

3.3 Applying GSP to Course-Sequence Data

When we run the GSP algorithm on course-sequence data at the departmental level, we restrict the data to only include sequences that contain students who take one or more courses from that department. This improves the algorithm's running time without impacting the generated frequent *k*-sequences; however, it impacts *minsup*, if expressed as a fraction, since the denominator used to calculate support is reduced. To address this, and also account for the fact that different departments have different numbers of students (i.e., sequences), we express *minsup* as a minimum support count (an integer) rather than a minimum support (a fraction).

The GSP algorithm can be applied directly to the student course-sequence data set, where each entry corresponds to the ordered courses taken by one student. If *minsup* is set too low, the number of frequent sequences will explode exponentially, which will radically impact the running time of the algorithm and lead to an unwieldy number of frequent sequences. Experiments in Section 4 vary *minsup* to assess its impact on running time and on the number of frequent sequences.

4. RESULTS

This section includes our main results. Section 4.1 provides department level information about the number of frequent *k*-sequences generated by the GSP algorithm, while Section 4.2 provides information on the running time of the algorithm. Section 4.3 examines the frequent sequences for the Computer Science (CS) department, introduces our course sequence flow diagram, and uses this diagram to provide insights into the sequencing of CS courses. The section concludes by analyzing some non-CS course sequences.

4.1 Summary Department Level Results

This section provides summary results of running the GSP algorithm on courses from individual STEM departments, combined Biology and Chemistry courses, and all undergraduate courses. By varying the number of courses considered we demonstrate its impact on the number of frequent sequences and running time. Table 1 shows the number of *k*-sequences generated for these groupings, for various values of *k* and *minsup*. Two entries are provided for each *minsup* value: one without pruning and one with pruning. The pruned results remove any frequent *k*-sequence contained (i.e., with the courses in the same relative order) within a larger frequent sequence, since this information can be considered redundant. Pruned entries have a "*" next to the *minsup* value. When generating frequent sequences from all undergraduate courses, much larger *minsup* values are employed to keep the running time manageable and to avoid generating more sequences than could be analyzed.

Table 1. Department-level results for the GSP algorithm

Department	<i>minsup</i>	Number of <i>k</i> -sequences (<i>k</i> from 2 to 9)							
		2	3	4	5	6	7	8	9
CompSci	50	151	223	132	23	1			
	50*	56	111	97	21	1			
	100	61	61	18	1				
	100*	61	61	18	1				
Chemistry	50	101	289	408	224	48	4		
	50*	9	44	178	152	40	4		
	100	20	7	1					
	100*	12	4	1					
Physics	50	51	48	37	21	7	1		
	50*	17	14	12	10	5	1		
	100	14	2						
	100*	10	2						
Biology	50	91	148	96	25	1			
	50*	22	61	59	23	1			
	100	47	54	26	6				
	100*	14	24	16	5				
NatSci	50	121	315	580	736	704	490	225	60
	50*	24	53	136	189	230	232	147	50
	100	70	124	131	86	34	4		
	100*	15	31	44	43	27	4		
Math	50	111	177	79	6				
	50*	40	98	68	6				
	100	68	34	1					
	100*	35	32	1					
Psychology	50	241	250	48					
	50*	110	193	48					
	100	114	53	8					
	100*	72	41	8					
Bio+Chem	50	278	899	1455	1272	694	235	38	
	50*	38	179	512	592	395	170	38	
	100	122	296	414	344	162	37	3	
	100*	48	115	202	248	174	64	3	
All	500	919	1853	1792	1096	532	197	47	5
	500*	322	819	902	599	328	141	40	
	1000	292	416	160	6				
	1000*	105	238	148	6				

* frequent *k*-sequences that appear in frequent *k*+1 sequences are pruned

Table 1 shows that the number of k -sequences varies greatly by department. This variation is due in part to the varying number of students, and student majors, in each department, as well as the amount of sequencing in the curriculum. The largest k -sequence for each department varies from 4 to 10 (Natural Science has one 10-sequence for $\text{minsup}=25$ that did not fit in the table). The number of k -sequences is also dramatically impacted by the minsup value, which indicates that care must be chosen in selecting this value. Adding additional courses can also have a large impact, as can be seen by comparing the individual Biology and Chemistry department entries with the combined Bio+Chem entry. The number of entries when using all courses would be enormous if we did not increase the minsup threshold by a factor of 10. The number of sequences that are pruned is more substantial for smaller sequences—a pruned itemset has a cascading effect as it trickles down to the lower levels (i.e., lower k values).

4.2 Run-time Complexity

The running time of the GSP algorithm is affected by the number of items (distinct courses), the number of sequences to mine, and the number of items (i.e., courses) in each sequence. While worst case performance is exponential in the number of items/courses, the performance is generally much better, depending on the minsup value. If minsup is sufficiently low, the complexity can even be linear in the number of sequences. The time to generate the frequent course sequences for the same nine department groups appearing in Table 1, when run on a MacBook Air laptop with an 8-core 3.2GHz processor, is displayed in Figure 1. To make the figure easy to read the legend is ordered to match the running times associated with each department. Since the y-axis uses a log-scale, the running time is not actually linear. In order to keep the running time manageable, minsup starts at 500 when mining all courses—but as shown in Table 1, this still yields a much longer running time.

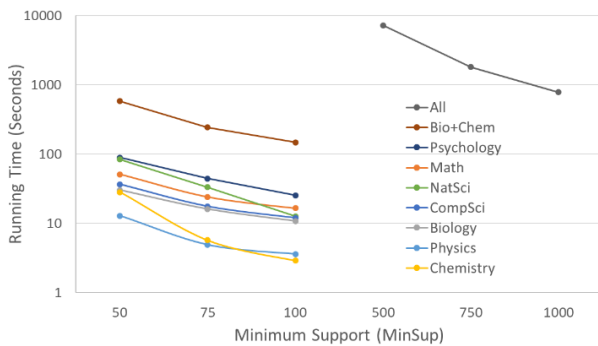


Figure 1. Impact of minimum support on running time

4.3 Detailed Analysis of Computer Science

This section looks into the detailed results for the CS department. In Section 4.3.1 we introduce our course sequence flow diagrams, which are capable of representing the information from a large number of frequent sequences in compact visual form. In Section 4.3.2 we then use this diagram to better understand how students sequence their CS courses.

4.3.1 Course Sequence Flow Diagram

Many of the results for the department are summarized in Figure 2, which embodies the information within all twenty-three Computer Science 5-sequences (the 5-sequences are listed in Appendix Table 2). We focus on these since there is only one 6-sequence and no larger sequences (see Table 1). Our course sequence flow diagram constitutes a substantial improvement upon previously existing sequencing visualization methods, such as Sankey diagrams [8]. We

believe that our course sequence flow diagram is a significant contribution, preserving information across multiple sequences while conveying this information clearly and compactly.

Figure 2 contains all eleven courses that are represented in the 23 5-sequences in Table 2. Each sequence in Table 2 has an index, and each node in the Figure 2 is labeled with the matching index, so the 23 5-sequences can be reconstructed from the diagram. For example, if you follow the flow of all nodes in Figure 2 that contain index 1, you will reconstruct the first sequence in Table 2 (DISC, CS1, CS2, DS, TOC). The diagram actually goes much further, however, presenting support information for each sequence. Each of the 23 sequences has its own support; when sequences overlap, the corresponding supports are added together in the current flow diagram. Thus, we can determine the combined support associated with any edge in the diagram based on the summed supports assigned to that edge. Appendix Table 3 provides the underlying information about each pair of courses across the extracted sequences. To avoid overcrowding the diagram with information, we encode the support sum from each edge through varying thicknesses. The thickness level listed in Table 3 and utilized in Figure 2 is computed as follows:

$$\text{Thickness} = \lfloor 2 * \log_{10} \Sigma \text{Support} \rfloor$$

4.3.2 Specific Insights into CS Based on Results

Figure 2 provides a succinct high level view of the flow of courses, which enables us to learn about how students sequence their courses. Given our experience with the undergraduate CS curriculum, we are able to verify known patterns in course sequences (first three bullet items) and glean new and potentially useful insights (last three bullet items). We summarize some of these as follows:

- The diagram recognizes the key prerequisite relationships including our introductory programming sequences of CS1 → CS2 → Data Structures (DS). These courses are connected with the thickest edges.
- Computer Organization is often taken after CS2 and Data Structures, which is what we recommend, but it is not required (it can be taken after CS1).
- The nodes in Figure 2 are arranged into 6 vertical columns based on the edges and relative position of the courses within the frequent sequences. These reflect the general structure and course levels quite well. For example, Discrete Structures and CS1 are 1000 level courses, CS2 and Data Structures are 2000 level courses, the courses in the next column are all 3000 level courses, etc. A few notable exceptions are discussed next.
- The position of Data Mining (DM) is surprising. Its only prerequisite is CS1, while Theory of Computation (TOC) has many prerequisites—yet Data Mining is generally taken after Theory of Computation. This may be an artifact of our assigning Data Mining as a 4000 level course even though our faculty believe that level is not appropriate.
- Theory of Computation is almost always taken after Algorithms (ALG), even though both are 4000-level courses with no prerequisite relationship. The observed sequencing could be an artifact of scheduling but may be related to student belief that the algorithms course is essential for obtaining employment, including internships. Relevant 2-tuples show that ALG→TOC occurs 2.4 times as often as the reverse ordering.
- Database Systems (DB) is taken a bit later in the sequence than we would expect and perhaps recommend. It only requires CS1 but is taken after CS2 and Data Structures.

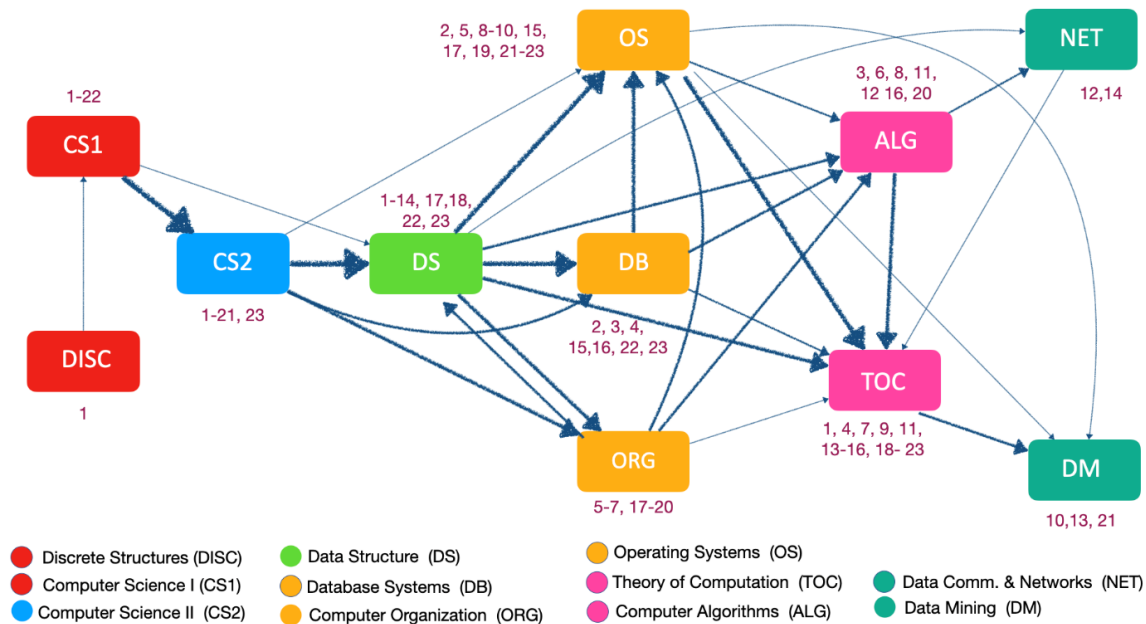


Figure 2. Course sequence flow diagram for Computer Science frequent 5-sequences (nodes labeled with indices from Table 2)

4.4 Analysis of Non-CS Departments

The dataset contains courses from dozens of departments. We lack the space and domain knowledge to analyze all of them, but we mine course sequences from several groupings of departments and provide a few observations. We start with the longest frequent sequences, which occur when courses from all departments are considered. With $minsup = 500$ we obtain five 9-sequences, all of which include only biology and chemistry courses, and four of the five sequences begin with the following eight courses:

IntroBio1+lab, GenChem1+lab, IntroBio2+lab, GenChem2+lab

Half of the courses above are lab courses which count as separate courses but probably should be excluded from the data set in order to find more meaningful sequences (we removed all lab courses for Computer Science only). Nonetheless the algorithm was able to identify a very common science sequence that is taken by students planning to go on to medical school. If we start to look at smaller k -sequences, such as when $k=5$, the sequences with the highest levels of support are dominated by courses that satisfy core requirements. In general we see that science courses are more tightly sequenced than other courses, although for shorter sequences the more popular core courses play a central role.

Due to their role in our university, we next present the results associated with core curriculum courses, which must be taken by all students, and span many departments. Below are the five 6-sequences that appear with $minsup = 1000$. These provide insight into the most common ways in which the core courses are taken:

IntroBio1+lab, GenChem1+lab, IntroBio2+lab
 MacroEcon, BusinessMath, MicroEcon, EngComp2, Business Calculus
 EngComp2, Philosophy1, Theology1, EngLit, Philosophy2
 EngComp2, Theology1, EngLit, Philosophy1, Philosophy2
 Phil1, EngComp2, Philosophy1, EngLit, Philosophy2
 Theology1, EngComp2, Philosophy1, EngLit, Philosophy2

The first two sequences are associated with science and business majors, respectively, who are directed to take those courses to fulfill their science, math, and social science core requirements; the last four are humanities courses taken by a wide range of students.

Given the Computer Science discipline has a close relationship with Mathematics, we also looked at the frequent sequences for the Math department. We were surprised to note that the last course in the sequence is very often Statistics, which we thought would be taken earlier, especially since courses with a higher number were sometimes taken first. But the department has the following long prerequisite sequence, which explains the observed behavior:

Calc1 \rightarrow Calc 2 \rightarrow Multivariable Calc \rightarrow Probability \rightarrow Statistics

5. CONCLUSION

This paper introduces GSP-based sequence mining for analyzing university course sequences. The method was applied to eight years of undergraduate course enrollment data and produced a large number of frequent k -sequences. Course sequences were generated for individual departments, several related departments, and across all departments. Our results show that it is possible to mine course sequences on a standard laptop given reasonable $minsup$ values.

We also developed a new course sequence flow diagram that visually captures many different common course sequences, while maintaining almost all of the low level information. We view the development of this diagram as a significant contribution since it is far superior to our prior efforts, and the Sankey diagrams [8] we generated using existing libraries. The course sequence flow diagram generated for Computer Science accurately represented well known course relationships and uncovered subtle sequencing issues. These insights can lead to improved advising and changes to our curriculum—for example, we can update course numbers to better reflect the role of the course, or to modify student behavior to better align with our sequencing intentions. Our analysis also provided insights into other course sequences and identified a long sequence taken by students in the pre-medical education track.

We believe that course sequencing mining can be a useful tool for academic advising and for better understanding how students sequence their courses. This work is an example of descriptive data mining and additional applications for this work will likely be discovered in time.

6. REFERENCES

- [1] Agrawal, R., and Srikant, R. 1994. Fast algorithms for mining association rules. In *Proceedings of the 20th International Conference on Very Large Data Bases, VLDB*, Vol. 1215, 487-499.
- [2] Betancur, L., Rottman, B. M., Votruba-Drzal, E., and Schunn, C. 2019. Analytical assessment of course sequencing: The case of methodological courses in psychology. *Journal of Educational Psychology*, 111(1), 91.
- [3] Borgelt, C. 2005. An Implementation of the FP-growth Algorithm. In *Proceedings of the 1st International Workshop on Open Source Data Mining: Frequent Pattern Mining Implementations*, 1-5.
- [4] Gutenbrunner, T., Leeds, D.D., Ross, S., Riad-Zaky, M., and Weiss, G.M. 2021. Measuring the academic impact of course sequencing using student grade data. In *Proceedings of The 14th International Conference on Educational Data Mining*, International Educational Data Mining Society, Paris France, June 29-July 2, 799-803.
- [5] Morsy, S., and Karypis, G. 2019. Will this course increase or decrease your GPA? Towards grade-aware course recommendation. *Journal of Educational Data Mining*, 11(2):20-46, 2019.
- [6] Parks, M. R., Faw, M., and Goldsmith, D. 2011. Undergraduate instruction in empirical research methods in communication: Assessment and recommendations. *Communication Education*, 60, 406-421. DOI= 10.1080/03634523.2011.562909
- [7] Richards, A. S. 2012. Course sequencing in the communication curriculum: A case study. *Communication Education*, 61(4), 395-427. DOI= 10.1080/03634523.2012.713500
- [8] Schmidt, M. 2008. The Sankey diagram in energy and material flow management: part II: methodology and current applications. *Journal of Industrial Ecology*, 12(2), 173-185.
- [9] Srikant, R., and Agrawal, R. 1996. Mining sequential patterns: Generalizations and performance improvements. In *International conference on extending database technology*, 1-17. Springer, Berlin, Heidelberg.

APPENDIX

Table 2. Computer Science course sequence table

Index	Computer Science Frequent 5-Sequence	Support	Index	Computer Science Frequent 5-Sequence	Support
1	Discrete Struct, CS1, CS2, Data Struct, Theory of Comp	50	13	CS1, CS2, Data Struct, Theory of Comp, Data Mining	63
2	CS1, CS2, Data Struct, Databases, Operating Sys	67	14	CS1, CS2, Data Struct, Data Comm and Net, Theory of Comp	55
3	CS1, CS2, Data Struct, Databases, Comp Alg	60	15	CS1, CS2, Databases, Operating Sys, Theory of Comp	80
4	CS1, CS2, Data Struct, Databases, Theory of Comp	73	16	CS1, CS2, Databases, Comp Alg, Theory of Comp	58
5	CS1, CS2, Data Struct, Comp Org, Operating Sys	52	17	CS1, CS2, Comp Org, Data Struct, Operating Sys	54
6	CS1, CS2, Data Struct, Comp Org, Comp Alg	60	18	CS1, CS2, Comp Org, Data Struct, Theory of Comp	52
7	CS1, CS2, Data Struct, Comp Org, Theory of Comp	52	19	CS1, CS2, Comp Org, Operating Sys, Theory of Comp	73
8	CS1, CS2, Data Struct, Operating Sys, Comp Alg	70	20	CS1, CS2, Comp Org, Comp Alg, Theory of Comp	59
9	CS1, CS2, Data Struct, Operating Sys, Theory of Comp	110	21	CS1, CS2, Operating Sys, Theory of Comp, Data Mining	50
10	CS1, CS2, Data Struct, Operating Sys, Data Mining	61	22	CS1, Data Struct, Databases, Operating Sys, Theory of Comp	56
11	CS1, CS2, Data Struct, Comp Alg, Theory of Comp	84	23	CS2, Data Struct, Databases, Operating Sys, Theory of Comp	56
12	CS1, CS2, Data Struct, Comp Alg, Data Comm and Net	55			

Table 3. Computer Science sequence edge table

C1→C2	Indices	Σ Support	Thickness	C1→C2	Indices	Σ Support	Thickness
CS1→CS2	1-21	1338	6	DB→ALG	3,16	118	3
CS2→DS	1-14, 23	968	6	TOC→DM	13,21	113	3
OS→TOC	9,15,19, 21-23	425	5	ORG→DS	17,18	106	3
DS→DB	2,3,4,22,23	312	5	ALG→NET	12	84	2
DS→OS	8-10,17	295	5	DB→TOC	4	73	2
DB→OS	2,15,22,23	259	4	OS→ALG	8	70	2
CS2→ORG	17-20	238	4	OS→DM	10	61	1
ALG→TOC	11,16,20	201	4	CS1→DS	22	56	1
DS→TOC	1,13,18	165	4	DS→NET	14	55	1
DS→ORG	5--7	164	4	NET→TOC	14	55	1
CS2→DB	15,16	138	3	ORG→TOC	7	52	1
DS→ALG	11,12	139	3	DISC→CS1	1	50	1
ORG→OS	5,19	125	3	CS2→OS	21	50	1
ORG→ALG	6,20	119	3				

Table 3 shows CS1→CS2 and CS2→Data Structures have the two highest support sums; this is reflected in the edge-widths in Figure 2.

Employing Tree-based Algorithms to Predict Students' Self-Efficacy in PISA 2018

Bin Tan
University of Alberta
btan4@ualberta.ca

Maria Cutumisu
University of Alberta
cutumisu@ualberta.ca

ABSTRACT

Self-efficacy is a critical psychological construct that has a substantial impact on students' learning experience and global well-being. Thus, the early identification of low self-efficacious learners is an important task for educators and researchers. This study uses machine learning (ML) approaches to model the self-efficacy of over 520,000 students based on their test performance and responses to survey questions in the Programme for International Student Assessment (PISA) 2018. Two tree-based ensemble learning models (random forest and XGBoost) were built using 64 predictors and evaluated using nested cross-validation with a grid search method. The results showed that, although both algorithms predicted self-efficacy accurately, XGBoost slightly outperformed Random Forest (RF). The findings also revealed that students' non-cognitive constructs such as meaning in life and the motivation for mastering tasks were the most important predictors. Theoretical contributions include the expansion of the body of literature on ML applications that predict students' self-efficacy and the potential advancement of theoretical models of self-efficacy. Practical contributions include the applications of tree-based algorithms to identify low self-efficacious individuals at scale, in a large international assessment. Implications include the development of systems that use ML algorithms to detect low self-efficacious learners and provide support for early interventions.

Keywords

Self-efficacy, large-scale assessment, PISA 2018, students' traits

1. INTRODUCTION

Self-efficacy represents individuals' general beliefs about their competencies of performing specific tasks or achieving goals [4]. Students' self-efficacy has been consistently associated with their learning achievement [3, 11, 14]. Students with higher self-efficacy, at all levels of competency, are more successful in school activities and use more effective learning strategies [21]. In addition, various empirical findings showed that self-efficacy is associated with academic engagement [25]. High self-efficacious students tend to report a higher level of academic aspirations, spend more time on homework, and gain more positive learning experiences [6]. Those students are more gratified and satisfied with their accomplishments [27]. Moreover, self-efficacy is highly linked to students' global well-being and life outcomes [11]. It has been

found that students with low self-efficacy are more likely to drop out of school, which jeopardizes their future employment prospects [5]. In addition, low self-efficacious students tend to suffer from many mental and behavioral problems such as depression [2], suicidal ideation and attempts [25], social avoidance [24], and addictive behaviors [23]. Thus, students' self-efficacy is an important topic for psychological and educational research. If students' self-efficacy can be screened and predicted, practitioners may be able to deliver early intervention to help low self-efficacious students improve their learning experience, global well-being, and life outcomes.

In this present study, two decision tree-based algorithms (RF and XGBoost) were trained based on over 520,000 students' test performance and responses to survey questions in PISA 2018. The proposed research questions (RQ) were:

- (1) Is it possible to use the RF and XGBoost algorithms to predict students' self-efficacy with a small error rate?
- (2) What are the most important predictors of self-efficacy in these models?

2. RELATED WORK

Self-efficacy has been increasingly used as a predictor in ML models. However, to date, despite the significance of self-efficacy for students' learning and life, there are very few studies that treated self-efficacy as the focal variable to be predicted. The first such study [17] used Naive Bayes and decision tree algorithms to generate two sets of classification models of self-efficacy (high vs. low). The first set of models were built based on the demographic factors of the students, whereas the other set of models added additional predictors that were obtained when students were exposed to an intelligent problem-solving tutoring system including biofeedback signals and recorded log data. The classification accuracy of the models ranged from 82.1% to 87.3%.

Later, a K-medoids clustering algorithm was employed to group similar students based on their gender, survey-reported self-efficacy, and collected natural language utterances during dialogue in an intelligent tutorial dialogue system [9]. Results revealed differences in the use of utterances between students with high and low self-efficacy. For example, students with high self-efficacy tend to use more confident utterances to express their understanding of the knowledge, compared to students with low self-efficacy who usually make less confident utterances.

Recent efforts have examined domain-specific self-efficacy. A study trained a K-nearest neighbor algorithm to classify 127 students' responses as low, middle, or high using a 21-item self-efficacy survey [1]. The optimal results of the model performance based on validation-set approach reached 92.3%. Another study applied a decision tree classifier to a dataset containing 1894

B. Tan and M. Cutumisu. Employing tree-based algorithms to predict students' self-efficacy in PISA 2018. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 634–639, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852970>

undergraduate students' survey data, obtaining the highest accuracy score of 82.58% [26].

3. METHODS

3.1 Data Source

The international large-scale dataset used in this study contained students' self-reported survey data and their test results of the OECD's PISA 2018. The dataset was publicly accessible at [20]. All students participating in PISA 2018 were included in this study regardless of their country of citizenship or origin. This constituted the original sample of 612,004 students from 74 countries and regions.

3.2 Tree-based Algorithms

We employed two tree-based algorithms to predict students' self-efficacy. Tree-based algorithms use a series of if-then rules to generate predictions. In each step of the series, the if-then rules separate data points into subsets according to a node where the prediction has the lowest error rate. By repeating the step, the split will eventually terminate when reaching the stopping criterion. Although singular tree models can be interpreted straightforwardly and work well with nonlinear relationships between predictors and the target variable, they usually have weaker predictive performance given that they are prone to overfitting, a situation where the supervised learning model fits too close to the training data to be able to generalize well and predict future data.

Tree-based ensemble learning methods are alternatives to singular decision trees by combining decision trees. The algorithms used in this study are RF and XGboost. RF is an ensemble learning algorithm developed based on two algorithms: decision tree and bootstrapping. Bootstrapping resamples data with replacement and it is used to repeatedly split the same dataset into bootstrapped samples based on which multiple decision trees can be built. Each of the trees built can generate a result. Then the algorithm makes the final decision by aggregating the results of all singular trees. In predicting the numerical values, the final result is calculated by averaging the results of all individual trees. The advantage of RF is that it is less prone to overfitting, which lifts the accuracy and stability of prediction to a much higher level.

XGBoost is another ensemble learning algorithm based on decision trees [7]. In contrast to RF, XGBoost employs boosting, a technique of correcting the errors of existing models by adding new models sequentially to predict the residuals of the existing model and, then, along with the existing model make another prediction. Through sequential iterations, each execution is completed on the same dataset and later models are improvements of prior models. Eventually, the errors will be gradually minimized; the algorithm stops when the model performance converges to a stable state.

3.3 Focal Variables

In this study, self-efficacy is the response or target variable (i.e., the variable to be predicted in the current supervised learning task). In the survey of PISA 2018, students' self-efficacy was measured by five items, namely, "I usually manage one way or another", "I feel proud that I have accomplished things", "I feel that I can handle many things at a time", "My belief in myself gets me through hard times", and "When I'm in a difficult situation, I can usually find my way out of it". Available responses were "Strongly disagree", "Disagree", "Agree", and "Strongly agree".

The predictors in the current study are the variables collected in the mandatory parts of students' self-reported questionnaire and their

test results in PISA 2018, classified as home factors, students' well-being, motivational factors, other non-cognitive constructs, school climate, teacher-related variables, personal experiences, as well as the PISA 2018 test performance [19]. Table 1 lists the predictors included and their dimensions. The reliability of students' self-reported scale scores was available in PISA technical reports [18]. All scales achieved at least an acceptable reliability.

Table 1: Predictors used in the tree-based models

Dimension	Variable name
Home factors	Home possessions
	Parents' professions and qualifications
	Parents' education backgrounds
	Parental support
School climate	Cooperation climate
	Disciplinary climate
	Competition climate
Teachers	Teacher support
	Teacher understanding
	Adaptive instruction
	Teacher feedback
	Teacher enthusiasm
Well-being	Teacher directed instruction
	Meaning in life
	Life satisfaction
	Positive affective states
	Lively
	Miserable
	Proud
	Afraid
	Sad
	Fear
Sacred	
Motivational factors	Learning interests
	Learning aspiration
	Value of school
	Motivation for mastering tasks
	Motivation for competition
Other non-cognitive constructs	Reading self-concept
	Fixed mindset
	Empathy
	Attitude toward bullying
	Sense of belonging
Personal experiences	Exposure to bullying
	Skipped class or being late
	The age of early childhood education
	The age of pre-primary education
	Grade repetition
PISA 2018 test performance	Reading performance
	Math performance
	Science performance
Other	Gender

3.4 Data Preprocessing

A two-stage method was adopted to deal with missing values. In the first step, the entire row of the data entry was excluded if there were missing data on any of the five items measuring self-efficacy. Listwise deletion was used because it does not introduce new errors to the outcome variable as replacing the missing data with other values. This step excluded 84,179 instances, so 527,825 instances remained. In the second step, the missing values of the predictors were replaced with their column medians.

First, the responses of reverse worded items were reverse coded. Second, if the predictor is a categorical variable and is not grouped

with other predictors (single-item scale), k-1 dummy variables (k is the number of categories) were created to replace the original categorical variable. Third, for multi-item scale response data (e.g., self-efficacy, measured by five items), a polytomous item response theory (IRT) model, the generalized partial credit model (GPCM) was used to transform the data to IRT scores ranging from negative to positive. In this way, self-efficacy scores became truly continuous data. This is a similar method that was adopted in the PISA 2018 Technical Report [18]. In order to facilitate meaningful interpretations, the IRT scores were linearly transformed using a formula: $X' = X \times 15 + 100$. The choice of multipliers for mean and standard deviation was arbitrary, just for the ease of interpretation (i.e., no negative self-efficacy scores). Such transformation does not alter the true comparative values of the measured constructs.

Upon the completion of data preprocessing, the dataset contained 64 individual predictors (including coded categorical variables) and one target variable (self-efficacy).

3.5 Model Training, Validating, and Testing

The *xgboost* and *scikit-learn* libraries in Python 3.7 were used to build the XGBoost and RF regressors. Model training, validating, and testing were conducted using the *scikit-learn* library. A nested cross-validation with grid search algorithm was used to obtain a robust and trustworthy estimation of the model tuning and performance [16]. As shown in Figure 1, the nested cross-validation algorithm has two layers: an outer three-fold cross-validation and an inner three-fold cross-validation. There was a total of nine distinct folds of inner cross-validation and three folds of outer cross-validation. The goal of the inner cross-validation was to find the hyperparameters yielding the best model performance, while the outer cross-validation was to test the generalizability of the tuned model performance to a new dataset.

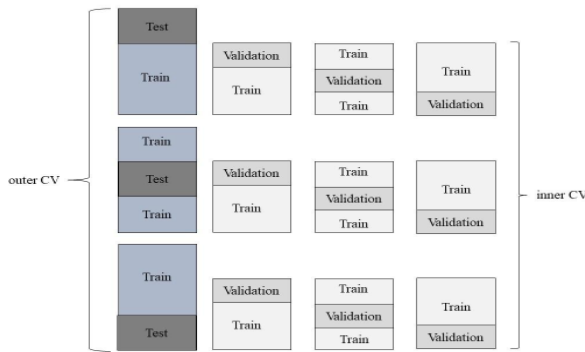


Figure 1: Nested cross-validation

Hyperparameters tuned for both RF and XGBoost regressors included the number of trees (*n_estimators*) and the maximum depth of the trees (*max_depth*). The maximum depth was selected in a range of 5 to 25, with a step of 5, whereas the number of trees could be 100, 150, or 200. Mean absolute error (MAE), Root mean square error (RMSE), and R^2 were used as evaluation metrics for both model validation and testing.

4. RESULTS

4.1 Descriptive Statistics of Self-Efficacy

The mean self-efficacy score of this sample was 99.99, ranging from 57.60 to 128.32, with a standard deviation of 13.28. Figure 2 shows the histogram of self-efficacy scores that indicates a

leptokurtic distribution. Thus, more students scored extremely high or low compared to a normal distribution.

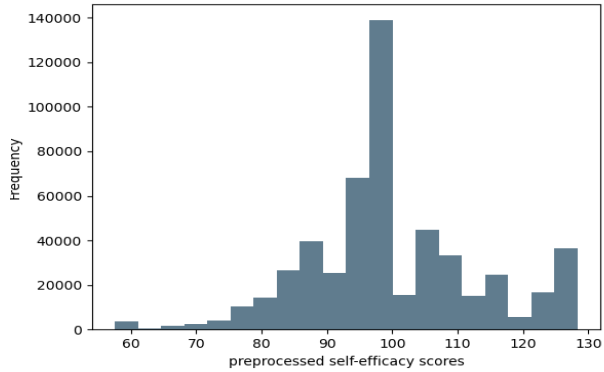


Figure 2: The distribution of self-efficacy IRT scores

4.2 Evaluation Results

Table 2 shows the training, validation, and test accuracy, for each set of the RF and XGBoost models, respectively. On the test set, R^2 of both prediction models was at least 0.447, suggesting that the two tree-based learning models could explain nearly half of the variability in students' self-efficacy. With reference to the range and standard deviation of self-efficacy scores, the MAEs and RMSEs indicated that both trained models achieved reliable prediction results.

Table 2: Results of model performance in training

Model	Data	RMSE	MAE	R^2
RF	Training set	4.240	3.279	0.898
	Validation set	9.898	7.373	0.444
	Test set	9.878	7.354	0.447
XGBoost	Training set	<.001	<.001	1
	Validation set	10.760	8.030	0.344
	Test set	9.776	7.271	0.458

4.3 Relative Importance of Predictors

The importance of the 64 predictors of students' self-efficacy was ranked. Figure 3 and Figure 4 present the top ten predictors and their weight contribution to the predictive power for the two models. In the RF model, the motivation for mastering tasks appeared to be the most powerful predictor with a relative importance of 19.5%, followed by meaning of life (10.8%), reading self-concept (5.1%), learning aspiration (4.4%), motivation for competition (3.2%), positive emotions (3%), empathy (2.9%), always feel proud (2.5%), and fear (2.5%).

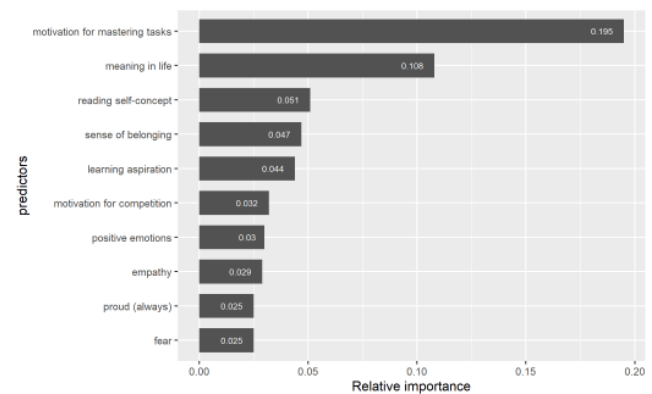


Figure 3: Relative predictor importance of the RF model

The ten most important predictors of the XGBoost model were the motivation for mastering tasks (14.5%), meaning in life (11.1%), proud (sometimes, always, rarely, and never; a total of 26.5%), motivation for competition (4%), learning aspiration (3.9%), positive emotions (3.2%), and reading self-concept (3.1%). These predictors contributed a total of 66.3% of the model prediction power. Notably, numerous variables were ranked in the top ten in both RF and XGBoost models, with motivation for task mastery and purpose in life maintaining the top two positions in both models. However, all the highly ranked predictors in both models appeared to be students' non-cognitive constructs including well-being and motivational factors. There were no variables of home factors, school climate, teachers, experience, and PISA 2018 test performance in the top 10 list.

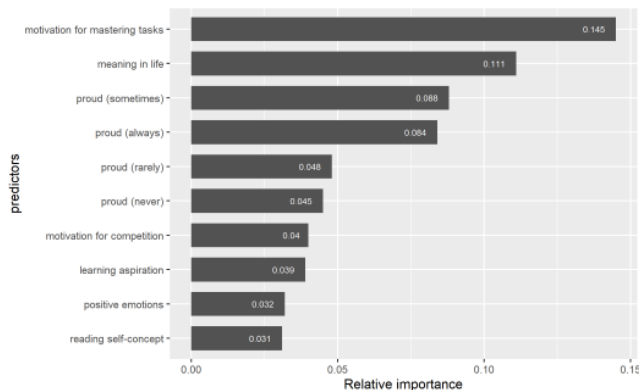


Figure 4: Relative predictor importance of the XGBoost model

5. DISCUSSION

The present study employed the RF and XGBoost algorithms to predict students' self-efficacy. The results suggest that the two tree-based algorithms could predict students' self-efficacy with small error sizes based on their self-reported survey data and test data. The XGBoost model seems to slightly outperform the RF model with respect to all chosen evaluation metrics on the test data.

The results also revealed the most salient predictors of both ML models. According to the rank of the relative importance, the best predictors for both models appeared to be students' non-cognitive factors including well-being and motivation. This is consistent with theories and empirical evidence [2, 13, 22, 27] supporting the close relationships between one's self-efficacy and their other non-cognitive constructs. On the other hand, gender was not a very important predictor. This is in line with a previous meta-analysis which reveals only a slight difference in self-efficacy between genders [13]. A surprising finding, however, is that students' test performances in PISA 2018 did not strongly predict self-efficacy. In a number of previous studies, researchers often consider self-efficacy as one of the strongest predictors for academic achievement [10]. However, this study revealed that predicting self-efficacy based on academic achievement seemed to be less unsuccessful. Another finding that is beyond our expectation is that home factors including parents' education and qualifications and home possessions contribute poorly to the predictive power in both models. This contradicts other studies which suggest the strong associations between self-efficacy and socioeconomic status [12, 15]. In addition, teachers-related variables and factors of school climate were not highly ranked predictors. We attribute the relatively weak predictive power of these predictors to their indirect relationship with students' self-efficacy. Home factors, academic achievement, as well as school and teacher factors, usually shape

students' self-efficacy through other non-cognitive constructs. This also explains why the non-cognitive constructs are better predictors in the current models.

The present study has three major implications. First, it provides a successful example of predicting students' self-efficacy, expanding the body of literature on self-efficacy modeling. Second, it ranks the relative importance of predictors for students' self-efficacy, paving the way for future studies to further examine the relationships between self-efficacy and its best predictors. This may advance theories of self-efficacy as such expanding the model of how one's self-efficacy is formed. Third, it predicts and models students' self-efficacy at scale, using data from an international assessment. Because high self-efficacy is beneficial to students' motivation and learning experience [8], while low self-efficacy is associated with many mental and behavioral problems [24], early identification of low efficacious students is critical to students' educational careers and global well-being. This study suggests that it is feasible for education systems to use ML approaches to identify low self-efficacious students at scale.

A limitation of this study is that the family-level, school-level, and country-level factors used to predict students' self-efficacy are not exhaustive in the current ML models. Although our findings indicate that factors such as students' home factors and learning environment have a negligible effect on the model's performance, the current study was not able to examine a number of other potentially significant features. For example, parenting styles may be predictive of students' self-efficacy at the family level; at the school level, the predictive effect of geography, socioeconomic position, and school resources remains unknown. Another limitation is that the dataset mainly relied on students' self-reported questionnaires. Due to the subjective nature of self-reported data, the quality of students' responses may be subjectively biased. Finally, more tuning is needed for these models to address the overfitting issue inherent with tree-based models.

6. CONCLUSION AND FUTURE WORK

Noting that very limited ML research has been conducted to model students' self-efficacy, this study is the first to establish tree-based models that successfully predict students' self-efficacy at a large scale. This study also identified important predictors of students' self-efficacy, which helps to identify students with low self-efficacy and develop targeted programs to potentially improve self-efficacy. In responding to the limitations of the current studies, future studies can seek to use a more comprehensive feature set that includes more family level, school level, and country-level variables. In addition, for objective and real-time monitoring of self-efficacy, future studies may use other methods to gather objective, real-time indicators for predicting students' self-efficacy. In the future, other ML algorithms (e.g., lasso, deep learning) will be employed to tackle this task. Thus, applying ML approaches to predicting students' self-efficacy is feasible and constitutes an important undertaking.

7. ACKNOWLEDGMENTS

We would like to thank the reviewers for their feedback. We are grateful to the following granting agencies for supporting this research: the Social Sciences and Humanities Research Council of Canada - Insight Grant (SSHRC IG) RES0048110 and the Natural Sciences and Engineering Research Council Discovery Grant (NSERC DG) RES0043209.

8. REFERENCES

- [1] Farooque, A., & Rizk, A. (2020). *Using k-nearest neighbors to classify undergraduate female self-efficacy in computer science*. [Poster Presentation]. Summer Undergraduate Research Fellowship, University of Houston.
- [2] Alexopoulos, G. S., Raue, P. J., Banerjee, S., Mauer, E., Marino, P., Soliman, M., Kanelopoulos, D., Solomonov, N., Adeagbo, A., & Sirey, J. A. (2021). Modifiable predictors of suicidal ideation during psychotherapy for late-life major depression. A machine learning approach. *Translational Psychiatry*, *11*(1), 1–8. <https://doi.org/10.1038/s41398-021-01656-5>
- [3] Areepattamannil, S., Freeman, J. G., & Klinger, D. A. (2011). Intrinsic motivation, extrinsic motivation, and academic achievement among Indian adolescents in Canada and India. *Social Psychology of Education*, *14*(3), 427–439. <https://doi.org/10.1007/s11218-011-9155-1>
- [4] Bandura, A. (1977). Self-efficacy: Toward a unifying theory of behavioral change. *Psychological Review*, *84*(2), 191–215. <https://doi.org/10.1037/0033-295X.84.2.191>
- [5] Bandura, A., Barbaranelli, C., Caprara, G. V., & Pastorelli, C. (2001). Self-efficacy beliefs as shapers of children's aspirations and career trajectories. *Child Development*, *72*(1), 187–206. <https://doi.org/10.1111/1467-8624.00273>
- [6] Bassi, M., Steca, P., Fave, A. D., & Caprara, G. V. (2006). Academic self-efficacy beliefs and quality of experience in learning. *Journal of Youth and Adolescence*, *36*(3), 301–312. <https://doi.org/10.1007/s10964-006-9069-y>
- [7] Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, (pp. 785–794). New York, NY, USA.
- [8] Chowdhury, M. S., & Shahabuddin, A. M. (2007). Self-Efficacy, Motivation and Their Relationship to Academic Performance of Bangladesh College Students. *College Quarterly*, *10*(1), 1-9.
- [9] Ezen-Can, A., & Boyer, K. E. (2014). Toward adaptive unsupervised dialogue act classification in tutoring by gender and self-efficacy. *Strategies*, *8*, 24.
- [10] Gabriel, F., Signolet, J., & Westwell, M. (2018). A machine learning approach to investigating the effects of mathematics dispositions on mathematical literacy. *International Journal of Research & Method in Education*, *41*(3), 306–327. <https://doi.org/10.1080/1743727X.2017.1301916>
- [11] Govorova, E., Benítez, I., & Muñoz, J. (2020). Predicting student well-being: Network analysis based on Pisa 2018. *International Journal of Environmental Research and Public Health*, *17*(11), 4014. <https://doi.org/10.3390/ijerph17114014>
- [12] Han, J., Chu, X., Song, H., & Li, Y. (2015). Social capital, socioeconomic status and self-efficacy. *Applied Economics and Finance*, *2*(1), 1–10.
- [13] Huang, C. (2013). Gender differences in academic self-efficacy: A meta-analysis. *European Journal of Psychology of Education*, *28*(1), 1–35. <https://doi.org/10.1007/s10212-011-0097-y>
- [14] Hwang, M. H., Choi, H. C., Lee, A., Culver, J. D., & Hutchison, B. (2015). The relationship between self-efficacy and academic achievement: A 5-year panel analysis. *The Asia-Pacific Education Researcher*, *25*(1), 89–98. <https://doi.org/10.1007/s40299-015-0236-3>
- [15] Karaarslan, G., & Sungur, S. (2011). Elementary students' self-efficacy beliefs in science: Role of grade level, gender, and socio-economic status. *Science Education International*, *22*(1), 72–79.
- [16] Krstajic, D., Buturovic, L. J., Leahy, D. E., & Thomas, S. (2014). Cross-validation pitfalls when selecting and assessing regression and classification models. *Journal of Cheminformatics*, *6*(1). <https://doi.org/10.1186/1758-2946-6-10>
- [17] McQuiggan, S. W., Mott, B. W., & Lester, J. C. (2008). Modeling self-efficacy in intelligent tutoring systems: An inductive approach. *User Modeling and User-Adapted Interaction*, *18*(1–2), 81–123. <https://doi.org/10.1007/s11257-007-9040-y>
- [18] OCED (2019). *PISA 2018 technical report*. OECD publishing.
- [19] OCED. (2019). *PISA 2018 assessment and analytical framework*. OECD publishing.
- [20] OECD (2019). *PISA 2018 Database* [Data set]. <https://www.oecd.org/pisa/data/2018database/>
- [21] Schunk, D. H., & Ertmer, P. A. (2000). Self-regulation and academic learning: Self-efficacy enhancing interventions. In M. Boekaerts, P. R. Pintrich, & M. Zeidner (Eds.), *Handbook of self-regulation* (pp. 631–649). Academic Press. <https://doi.org/10.1016/B978-012109890-2/50048-2>
- [22] Schunk, D. H., & Pajares, F. (2002). The development of academic self-efficacy. In *Development of achievement motivation* (pp. 15-31). Academic Press. <https://doi.org/10.1016/b978-012750053-9/50003-6>
- [23] Schwarzer, R., & Luszczynska, A. (2006). Self-efficacy, adolescents' risk-taking behaviors, and health. *Self-Efficacy Beliefs of Adolescents*, *5*, 139–159.
- [24] Tahmassian, K., & Moghadam, N. J. (2011). Relationship between self-efficacy and symptoms of anxiety, depression, worry and social avoidance in a normal sample of students. *Iranian Journal of Psychiatry and Behavioral Sciences*, *5*(2), 91.
- [25] Valois, R. F., Zullig, K. J., & Hunter, A. A. (2013). Association between adolescent suicide ideation, suicide attempts and emotional self-efficacy. *Journal of Child and Family Studies*, *24*(2), 237–248. <https://doi.org/10.1007/s10826-013-9829-8>
- [26] Wirojcharoenwong, W., Luangnaruedom, N., Rattanasiriwongwut, M., & Tiantong, M. (2014). Decision Tree Classifier for Computer Self-Efficacy with Best First Feature Selection. *International Journal of the Computer, the Internet and Management*, *22*, 62-67. <https://doi.org/10.3390/su13169337>
- [27] Zimmerman, B.J. (1995). Self-efficacy and educational development. In A. Bandura (Ed.), *Self-Efficacy in Changing Societies* (pp.202-231). New York: Cambridge University Press.

Identifying Longitudinal Attendance Patterns through Student Subpopulation Distribution Comparison

Zitong Zhao¹, Pan Deng², Jianjun Zhou²

¹The Chinese University of Hong Kong, Shenzhen, China

²Shenzhen Research Institute of Big Data, Shenzhen, China

{zitongzhao, pandeng}@link.cuhk.edu.cn

zhoujianjun@cuhk.edu.cn

ABSTRACT

Attendance reflects students' study motivation, and serves as an important indicator in educational management. A strong correlation has been found between attendance and academic performance. Due to the time varying nature of attendance and the challenge of collecting data automatically, few studies explored the longitudinal attendance of student subpopulations. In this paper, we introduce a new method combining Exponential Moving Average (EMA) and Kullback-Leibler Divergence (KLD) to identify longitudinal attendance patterns. We justify that KLD best preserves the structural difference between attendance distributions in student subpopulations. Using real-life data from a university, our result identifies the critical period when high and low academic performance students diverge in attendance, which calls for the attention of educators.

Keywords

longitude attendance data, exponential moving average, Kullback-Leibler divergence, behavior pattern

1. INTRODUCTION

Attendance is an important indicator in education. It interplays with many other factors, such as instructional quality, psychological status, and academic performance which lies in the essence of education. Many studies confirmed the correlation between attendance and attainment, using methods including hypothesis test [20], correlation analysis [1] and prediction [32, 33]. Furthermore, many studies identified positive relation between drop out and low attendance [12, 30]. These studies suggest that attention data provides valuable information in the early detection and assistance of students with low attendance. However, few studies considered associating longitudinal attendance data with academic performance.

Longitudinal attendance data reveals the dynamics between attendance and academic success, but is still under-researched. Though many educators are aware of the importance of attendance and have carried out certain attendance policy, the lack of

Z. Zhao, P. Deng, and J. Zhou. Identifying longitudinal attendance patterns through student subpopulation distribution comparison. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 640–646, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853034>

deep understanding of the interaction between attendance and academic success hinders the design and implementation of effective attendance policy. This is especially true for universities, where a mandatory attendance policy on every course at all time is usually not pervasive.

With the popularity of smart devices and the arrival of Internet of Things (IoT), longitudinal attendance data can be collected automatically and in a large scale. Techniques such as Bluetooth/Beacon [2, 31], face recognition [13] and Wi-Fi tracing [23, 34] have been applied to detect attendance automatically. [7] collected attendance data in an institution-wide fashion. The abundance of data makes it easier to study longitudinal attendance, and calls for new methods and in-depth analysis.

Traditionally longitudinal attendance data is assessed by simple statistics and line charts. The average attendance rate over a period of time (e.g., a week or a semester) is a widely adopted indicator of students' attendance. While being effective in many educational applications, a single average value is subject to noise in the period. For example, in attendance data collected using Wi-Fi tracing [7, 23, 34], the mal-function of some access points on campus can lead to inaccurate weekly attendance data for students. When studying the weekly attendance data in a time series, the fluctuation of data points can lead to wrong interpretation of the trend in attendance.

The comparison of the attendance for two student populations is another challenge. For example, it is interesting to compare the trends of attendance for students with top 25% and last 25% academic performance. However, the handy line chart of weekly average attendance rates does not fully reflect the structure inside each population.

Below we list the research questions of this study.

- 1) How to measure attendance dissimilarity from the fluctuating attendance time series between student subpopulations with different academic performance?
- 2) What can we learn from the divergence of attendance patterns for students with different academic performance?

For Question 1, when measuring the dissimilarity between two populations, in order to preserve as much information in the student populations as we can, we apply Kullback-Leibler divergence (KLD) on attendance data. KLD was first introduced in 1951 by Kullback and Leibler [18]. It was proposed as a measure of dissimilarity between statistical populations in terms of information. We will show below that comparing with other commonly used methods, KLD emphasizes more on inner struc-

tural divergence between the attendance data of student populations with different academic performance.

To deal with fluctuation, we propose to apply Exponential Moving Average (EMA) to smooth the time series in order to obtain trend and regional information. Moving average is a simple but widely adopted technique in time series analysis, and has been applied to process longitudinal data in many areas. Learning from the successful application of exponential moving average methods on implementing weekly stock index [9], we apply it on weekly attendance index calculation.

For Question 2, we propose the combination of Exponential Moving Average and KLD to detect critical periods when attendance patterns of students with top 25% and last 25% academic performance diverge dramatically.

In summary, the contributions of this study are as follows. We propose to apply the combination of Exponential Moving Average (EMA) and Kullback-Leibler divergence (KLD) to study longitudinal attendance data. So far as we know, there is no previous study that apply EMA or KLD on attendance data. We derive random subsamples to study moving average indexes and dissimilarity functions of student populations. Experiment results on attendance data shows that our method out-performs traditional methods. The application of our method on first year university students shows that the attendance of the top 25% and last 25% students on academic performance diverges during mid-term periods and holidays, which calls for the attention of university managements.

The rest of this paper is organized as follows. Section 2 describes the related studies. Section 3 introduces our method. Section 4 presents the results. Section 5 describes the limitations. Section 6 draws the conclusions.

2. RELATED STUDIES

The time variant nature of attendance contains rich details for student behaviors changes; however, before attendance data can be collected automatically, attendance is usually collected by sign-up sheets which have to be process manually and hinder the study of longitudinal attendance data. [6, 21] observed declines in student attendance over the duration of academic years. [12, 27] treated students' attendance as time series data and performed clustering. [32] studied the life-cycles of 48 students using multiple dimensions of data, including raw attendance data from GPS tracing and Wi-Fi tracing. Valuable information was collected from these results.

Critical period detection is often associated with time series analysis. [11] divided students into several bands according to their academic performance, plotted the cumulative credit time series of each student subpopulation and detected divergent critical points using unpaired t-test. [29] identified students' withdrawal routes by showing the proportion of students not submitting their assignments and found that only very few of them returning to submit the next ones. [4] researched into MOOC data, plotted students' activity time series and found activity peaks before each assessment due. None of these researches considered attendance time series. Using attendance records from high schools in New York, [16, 17] found weekly cycles and detected abnormal data points. The authors [15] then utilized the results to identify problems and provide suggestion for high school educators. These researches considered attendance, but did not associate it with academic performance.

Kullback-Leibler Divergence (KLD) is a popular method to calculate the difference between two populations. It has been justified as a sensitive dissimilarity measurement for probability distribution [35]. In the context of education, KLD is widely applied to detect test collusion by comparing unusual distribution of individual testing behavior to that of the whole test taker population [3, 19]. Besides, KLD was applied in [8, 26, 28] to classify students into groups by maximizing the divergence of certain quality between distributions of different subpopulations. [24] observed student's learning trajectory by calculating the difference between quizzes data collected before and after instructions. So far as we know, there is no previous study that applied KLD to attendance data of student populations.

Moving Average (MA) method is one of the most popular techniques in time series analysis. It's often used for the purpose to smooth and track the trend in longitudinal data [9]. Exponential Moving Average, as a variation of Moving Average, is widely used in many area, including finance [9, 10], sales analysis[14], weather forecasting[5] and education [25] (to predict enthusiasts in mathematics programs). So far as we know, no previous study has applied exponential moving average on attendance data.

3. METHOD

3.1 Data and Setup

Anonymous data of students' cumulative GPA (denoted as CGPA) and attendance data were retrieved from a university in China with approval from the university management. The data was collected from undergraduate students spanning over 3 semesters in 2018 and 2019, namely, Fall 2018, Spring 2019 and Fall 2019 (noted that academic year starts in a Fall semester). To protect privacy, student IDs were converted into hash code beforehand.

For most courses, students received letter grades from A to F. As an example, only 15 out of 426 courses taught in academic year 2018 are courses with P/F grades. These courses were mostly for internship or research methodology/seminar, which were seldom taken by undergraduates in their first two years of study. All courses with P/F grades were excluded in calculation.

Attendance data was collected using a Wi-Fi based method and was handled using method proposed in [23, 34]. Students' weekly attendance rate was calculated by taking the average of attendance within a week. For most courses, no mandatory attendance policy was established. The rest of courses that declared mandatory attendance policy were mostly university core courses which should be taken by all undergraduates.

3.2 Moving Average Index

In this subsection, we introduce exponential moving average methods to smooth the attendance curve.

Let a_t be the attendance rate at time t for student i , then we have $a_t \in [0, 1]$. Let $EMA_{i,t}$ be the corresponding index for student i at time t . Exponential moving average can be calculated using the formulas below.

Definition 1 Exponential Moving Average (EMA):

β stands for the weighting decay factor taken between 0 and 1. Commonly it's set to be $\beta = 2/(n + 1)$ with $n = 2$ in our experiment (see the appendix for details).

EMA assigns the highest weight for the most recent data. The weight for the earlier data points decay with time via recursive process. Therefore, the earliest data would never vanish in the process of gaining successive values.

3.3 Dissimilarity Measurement

In this subsection, we study the dissimilarity measurements between attendance data of two student populations. Ideally, we want to find a dissimilarity measurement that can best reflect inner structure divergence between the two concerned populations.

Definition 2 Mean Difference (MDF): Let $a_{i,t}$ be the attendance rate at time t for student i . Let P and Q be two distributions of $a_{i,t}$ at time t of two populations of students. The mean difference (MDF) of P and Q is calculated as below.

$$MDF(P, Q) = E(P) - E(Q)$$

MDF is the simplest way to measure difference between two population. However, this measurement ignores the inner structure of distributions.

Definition 3 Absolute Distance (AD) & Definition 4 Kullback-Leibler Divergence (KLD): We divide the domain of distributions P and Q into several attendance intervals and name these intervals $1, 2, \dots, n$. Then the probability of a student from distribution P or Q falling in a given interval is p_1, p_2, \dots, p_n or q_1, q_2, \dots, q_n . We calculate the absolute distance (AD) and KLD as follows.

$$AD(P, Q) = \sum_{j=1}^n |p_j - q_j|$$

$$KLD(P, Q) = \sum_{j=1}^n p_j \log \left(\frac{p_j}{q_j} \right)$$

Both AD and KLD consider the inner structures of the distributions [35]. However, KLD is more sensitive when detecting inner structure differences than absolute distance.

The difference between AD and KLD can be illustrated by an example as follows. Let A, B1, B2 be three student populations. Divide the domain evenly into 4 intervals based on the attendance rates of the students: last 25% quantile, 25% - 50%, 50% - 75%, top 25% quantile. Probability of the three populations fall into the above-mentioned intervals are listed below. For example, 0.25 represent 25% of the population.

A: [0.25, 0.25, 0.25, 0.25]

B1: [0.30, 0.30, 0.30, 0.10]

B2: [0.40, 0.25, 0.25, 0.10]

For A, the attendance rates of students are evenly distributed in the 4 intervals. B1 differs from A by moving 15% of the population from the top quarter to the other 3 intervals evenly. B2 differs from A by moving 15% of the population from the top quarter to the last quarter only. Therefore, A overall should be considered to have higher attendance than B1 and B1 higher than B2.

Calculate the dissimilarity between A and B1, A and B2 using AD and KLD respectively, we have

$$AD(A, B1) = AD(A, B2) = 0.3$$

$$KLD(A, B1) = 0.072, KLD(A, B2) = 0.096$$

$$KLD(A, B2) > KLD(A, B1)$$

In this example, AD cannot distinguish the difference between B1 and B2 in terms of the dissimilarity to A, while KLD correctly measures the dissimilarity order. It illustrates that KLD is more sensitive to inner structure differences than AD. MDF is not calculated in this example because we do not have the detailed distribution. In the result section, we will demonstrate the advantage of KLD over MDF and AD using random subsamples derived from real-life data.

4. RESULTS

4.1 Dissimilarity Measurement Selection

We make comparisons of dissimilarity measurements using large samples of random groups synthesized from our dataset. Though this experiment, we want to show that KLD can best illustrate the difference in attendance data between top students and other students.

Attendance data used is collected from students from Cohort 2018 in Fall 2018 and Spring 2019 semesters. We consider students' overall attendance rates in the academic year 2018-19. We use their first year CGPA as the measurement of academic performance.

There are 120 students whose academic performance are in top 10%. Therefore, we set the top 120 students as top 10% group and randomly generate 10000 groups, each of 120 randomly selected students from the whole student population.

We divide the whole student population evenly into 4 levels based on their overall attendance rates. In each level there are 25% of students from the whole population. we then calculate for each group, the portion of students in each level. For example, for top 10% group, its attendance rate distribution in the 4 levels (from low to high) is: [0.125, 0.242, 0.3, 0.333]

We perform the three candidate dissimilarity measurements between the 10000 groups and the top 10% group. Noted that for KL-divergence, the direction is from the top 10% group to the generated group. We then find out the 3 groups (named as KLD, AD, MDF) having the largest attendance dissimilarity via the 3 different measurements and rank the 3 groups' academic performance among the 10000 groups. Intuitively, the group with the largest difference from the top 10% group have the poorest academic performance.

Table 1. The rankings of different dissimilarity measurements

	Mean GPA	Mean GPA Rank	Last 10% Ratio	Last 10% Ratio Rank
Top 10%	3.77	10000	0	10000
KLD	3.02	1	14.17%	328
AD	3.14	2157	10.83%	2670
MDF	3.14	1714	10.00%	3759

The results are shown in Table 1. The rankings of mean GPA and the last 10% ratio indicate the academic performance of the chosen group among the 10000 groups (the higher the ranking the poorer the academic performance). We can see that the group detected by KLD has the highest mean GPA rank (last 1) among 10000 groups and the highest last 10% ratio rank (last 328) among the three dissimilarity measurements. Therefore, KLD is a good

dissimilarity measurement and we apply it for distribution comparison on attendance difference.

For more experiment of dissimilarity measurement selection, please see the appendix.

4.2 Exponential Moving Average and KLD

In this subsection, we compare the performance of two methods, KLD combined with EMA (denoted as EMA+KLD) and KLD combined with the raw attendance rates (denoted as RAW+KLD). The partition of data and the experiment setup are similar to those in section 4.1.

The results are shown in Table 2. The group detected by EMA combined with KLD has a higher mean GPA rank (last 24) and a higher last 10% ratio rank (last 276) than the group detected by Raw+KLD. Therefore, combining exponential moving average and KLD improves our ability to detect attendance divergence between student subpopulations of different academic performance.

Table 2. The rankings of two methods

	Mean GPA	Mean GPA Rank	Last 10% Ratio	Last 10% Ratio Rank
Top 10%	3.743	10000	0	10000
Raw+KLD	3.144	2085	9.65%	4801
EMA+KLD	3.063	24	14.91%	276

For more information of EMA, please see the appendix.

4.3 Critical Periods in Attendance Divergence

In this subsection, we discuss the longitudinal attendance divergence between top and last 25% students on CGPA. The data is collected from first year undergraduate students in Fall 2018 and Spring 2019.

Figure 1 shows the KLD time series calculated between attendance of top and last 25% students using weekly attendance indexes processed with EMA. Noted that for KLD, we calculated the distance in the direction from the top population to the last population. The top and last quarter students first diverge after the 3-days national holiday (Fall 2018 Week 4, the Mid-autumn festival). In the mid-term period, the difference enlarges again and the difference stays at about the same level till the end of the semester. At the beginning of Spring 2019 semester, the attendance difference is reduced but is restored to the level of the last term quickly.

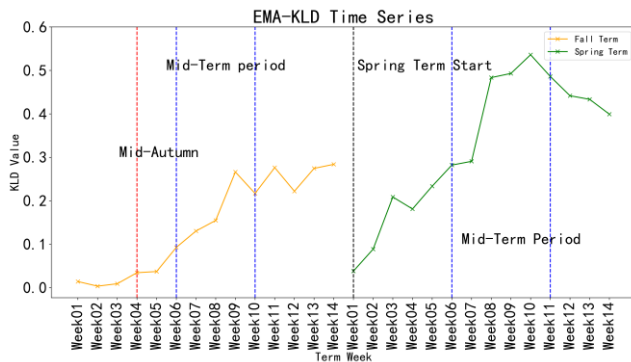


Figure 1. KLD time series.

During the mid-term of Spring 2019, the difference is enlarged and stays at the same level till the end of the semester.

We conjecture several reasons behind the divergence of attendance rates. In the local culture, the 3-days national holiday (Mid-autumn festival) is the time when people travel to meet their loved ones. Since the holiday is close in time to another national holiday, some students may leave the university and do not return to the university till the end of the second holiday. This behavior could lead to low attendance rates among the last quarter students during the week after the Mid-autumn festival. The mid-term period significantly enlarges the difference and multiple factors can be behind the phenomenon. The last quarter students could be frustrated by the mid-term exams and give up their studies; Preparing the mid-term exams is stressful and some students did not relax properly after the exams and start skipping classes; Some students might be addicted to video games and persistently reduce their class attendance.

5. LIMITATIONS

Raw attendance rate data used in our research was collected using a Wi-Fi based method proposed in [23, 34]. The data can be biased because some students would close their Wi-Fi connections or even close their digital devices all together, leading to a false label of absence. To avoid the possible bias, students whose data having less than 50 Wi-Fi connection records in a week were excluded. Previous studies [7, 22, 33] show that attendance collected from Wi-Fi tracing has usable accuracy. EMA can help reducing the noise and fluctuations in the attendance data.

Privacy is an importance issue in research related to Wi-Fi tracing. In our study, we removed personal information from the data and encrypted the student ID with a hash code. Attendance data and CGPA data were connected using this hash code. The Wi-Fi data collection had a clear location boundary and data was collected only when the students are on campus. We did not collect website URLs or communication content. Given that the research can potentially improve the existing attendance policy, this study has been approved by the university management.

6. CONCLUSIONS

In this study, we identified the divergence of longitudinal attendance patterns between student subpopulations of different academic performance using the combination of Exponential Moving Average (EMA) and Kullback-Leibler Divergence (KLD).

We designed several experiments to prove the efficacy of EMA and KLD to process raw attendance data and measure dissimilarity between top and last student populations. We then combined EMA and KLD to analyse real-life attendance data from two student subpopulations of top and last academic performance. The resulting curves are intuitive and imply rapidly increasing attendance divergence during the mid-term periods and right after public holidays.

Through the visualized results generated using our proposed method, we address the importance of longitudinal attendance patterns on academic performance. Our result gave educator an example on how to measure longitudinal attendance and can potentially help institutes to optimize attendance policy.

7. ACKNOWLEDGEMENTS

This work was supported by Shenzhen Research Institute of Big Data.

8. REFERENCES

- [1] Alexander, V. and Hicks, R.E. 2016. Does class attendance predict academic performance in first year psychology tutorials. *International Journal of Psychological Studies*. 8, 1 (2016), 28–32. DOI:<https://doi.org/10.5539/ijps.v8n1p28>.
- [2] Avireddy, S., Veerapandian, P., Ganapati, S., Venkat, M., Ranganathan, P. and Perumal, V. 2013. MITSAT — An automated student attendance tracking system using Bluetooth and EyeOS. *Proceedings of 2013 International Multi-Conference on Automation, Computing, Communication, Control and Compressed Sensing (iMac4s)* (2013), 547–552.
- [3] Belov, D.I. 2013. Detection of Test Collusion via Kullback–Leibler Divergence. *Journal of Educational Measurement*. 50, 2 (2013), 141–163. DOI:<https://doi.org/10.1111/jedm.12008>.
- [4] Breslow, L., Pritchard, D.E., DeBoer, J., Stump, G.S., Ho, A.D. and Seaton, D.T. 2013. Studying learning in the worldwide classroom research into edX’s first MOOC. *Research & Practice in Assessment*. 8, (2013), 13–25.
- [5] Cadenas, E., Jaramillo, O.A. and Rivera, W. 2010. Analysis and forecasting of wind velocity in chetumal, quintana roo, using the single exponential smoothing method. *Renewable Energy*. 35, 5 (2010), 925–930. DOI:<https://doi.org/10.1016/j.renene.2009.10.037>.
- [6] Colby, J. 2005. Attendance and Attainment—a comparative study. *Innovation in Teaching and Learning in Information and Computer Sciences*. 4, 2 (2005), 1–13. DOI:<https://doi.org/10.11120/ital.2005.04020002>.
- [7] Deng, P., Zhou, J., Lyu, J. and Zhao, Z. Assessing Attendance by Peer Information. *Proceedings of The 14th International Conference on Educational Data Mining (EDM21)* (Paris, France), 400–406.
- [8] Faucon, L., Olsen, J.K. and Dillenbourg, P. 2020. A Bayesian Model of Individual Differences and Flexibility in Inductive Reasoning for Categorization of Examples. *Proceedings of the Tenth International Conference on Learning Analytics & Knowledge (LAK20)* (New York, NY, USA, 2020), 285–294.
- [9] Hansun, S. 2013. A new approach of moving average method in time series analysis. *Proceedings of 2013 Conference on New Media Studies (CoNMedia)* (2013), 1–4.
- [10] Hansun, S. and Kristanda, M.B. 2017. Performance analysis of conventional moving average methods in forex forecasting. *Proceedings of 2017 International Conference on Smart Cities, Automation & Intelligent Computing Systems (ICONSONICS)* (2017), 11–17.
- [11] Hlosta, M., Kocvara, J., Beran, D. and Zdrahal, Z. 2019. Visualisation of key splitting milestones to support interventions. *Companion Proceeding of the 9th International Conference on Learning Analytics & Knowledge (LAK’19)* (Tempe, Arizona, USA, Mar. 2019), 776–784.
- [12] Hung, J.-L., Wang, M.C., Wang, S., Abdelrasoul, M., Li, Y. and He, W. 2017. Identifying At-Risk Students for Early Interventions—A Time-Series Clustering Approach. *IEEE Transactions on Emerging Topics in Computing*. 5, 1 (2017), 45–55. DOI:<https://doi.org/10.1109/TETC.2015.2504239>.
- [13] Kar, N., Debbarma, M.K., Saha, A. and Pal, D.R. 2012. Study of implementing automated attendance system using face recognition technique. *International Journal of computer and communication engineering*. 1, 2 (2012), 100.
- [14] Karmaker, C., Halder, P. and Sarker, E. 2017. A study of time series model for predicting jute yarn demand: case study. *Journal of industrial engineering*. 2017, (2017).
- [15] Koopmans, M. 2018. Exploring the Effects of Creating Small High Schools on Daily Attendance: A Statistical Case Study. *Complicity: An International Journal of Complexity and Education*. 15, 1 (2018), 19–30. DOI:<https://doi.org/10.29173/cmplct29352>.
- [16] Koopmans, M. 2016. Investigating the Long Memory Process in Daily High School Attendance Data. *Complex Dynamical Systems in Education: Concepts, Methods and Applications*. M. Koopmans and D. Stamovlasis, eds. Springer International Publishing. 299–321.
- [17] Koopmans, M. 2017. Nonlinear Processes in Time-Ordered Observations: Self-Organized Criticality in Daily High School Attendance. *Complicity: An International Journal of Complexity and Education*. 14, 2 (2017), 78–87. DOI:<https://doi.org/10.29173/cmplct29337>.
- [18] Kullback, S. and Leibler, R.A. 1951. On Information and Sufficiency. *The Annals of Mathematical Statistics*. 22, 1 (1951), 79–86.
- [19] Man, K., Harring, J.R., Ouyang, Y. and Thomas, S.L. 2018. Response Time Based Nonparametric Kullback-Leibler Divergence Measure for Detecting Aberrant Test-Taking Behavior. *International Journal of Testing*. 18, 2 (2018), 155–177. DOI:<https://doi.org/10.1080/15305058.2018.1429446>.
- [20] Massingham, P. and Herrington, T. 2006. Does Attendance Matter? An Examination of Student Attitudes, Participation, Performance and Attendance. *Journal of University Teaching and Learning Practice*. 3, 2 (2006), 83–103. DOI:<http://dx.doi.org/10.53761/1.3.2.3>.
- [21] Newman-Ford, L., Fitzgibbon, K., Lloyd, S. and Thomas, S. 2008. A large-scale investigation into the relationship between attendance and attainment: a study using an innovative, electronic attendance monitoring system. *Studies in Higher Education*. 33, 6 (2008), 699–717. DOI:<https://doi.org/10.1080/03075070802457066>.
- [22] Patel, A., Joseph, A., Survase, S. and Nair, R. 2019. Smart Student Attendance System Using QR Code. *Proceedings of The 2nd International Conference on Advances in Science & Technology (ICAST)* (Mumbai, India, Apr. 2019).
- [23] Pytlarz, I., Pu, S., Patel, M. and Prabhu, R. 2018. What Can We Learn from College Students’ Network Transactions? Constructing Useful Features for Student Success Prediction. *Proceedings of The 11th International Conference on Educa-*

- tional Data Mining (EDM 2018)* (Buffalo, New York, Jul. 2018).
- [24] Sapountzi, A., Bhulai, S., Cornelisz, I. and Klaveren, C. van 2021. Analysis of stopping criteria for Bayesian Adaptive Mastery Assessment. *Proceedings of The 14th International Conference on Educational Data Mining (EDM21)* (Paris, France, Jul. 2021), 630–634.
- [25] Setiawani, S., Fatahillah, A. and Lailani, P. 2020. Time series of mathematics education program, FKIP university of jember enthusiast through exponential smoothing method. *Journal of Physics: Conference Series*. 1465, 1 (Feb. 2020). DOI:<https://doi.org/10.1088/1742-6596/1465/1/012017>.
- [26] Shabaninejad, S., Khosravi, H., Indulska, M., Bakharia, A. and Isaías, P. 2020. Automated Insightful Drill-down Recommendations for Learning Analytics Dashboards. *Proceedings of the Tenth International Conference on Learning Analytics & Knowledge (LAK20)* (New York, USA, Jul. 2020), 41–46.
- [27] Sher, V., Hatala, M. and Gašević, D. 2020. Analyzing the Consistency in Within-Activity Learning Patterns in Blended Learning. *Proceedings of the Tenth International Conference on Learning Analytics & Knowledge (LAK20)* (New York, NY, USA, 2020), 1–10.
- [28] Shimada, A., Mouri, K., Taniguchi, Y., Ogata, H., Taniguchi, R. and Konomi, S. 2019. Optimizing Assignment of Students to Course based on Learning Activity Analytics. *Proceedings of The 12th International Conference on Educational Data Mining (EDM 2019)* (Montréal, Canada, Jul. 2019), 178–187.
- [29] Simpson, O. 2004. The impact on retention of interventions to support distance learning students. *Open Learning: The Journal of Open, Distance and e-Learning*. 19, 1 (2004), 79–95.
- [30] Tanvir, H. and Chounta, I.-A. 2021. Exploring the Importance of Factors Contributing to Dropouts in Higher Education Over Time. *Proceedings of The 14th International Conference on Educational Data Mining (EDM21)* (Paris, France, Jun. 2021), 502–509.
- [31] Varshini, A. and Indhurekha, S. 2017. Attendance system using beacon technology. *International Journal of Scientific & Engineering Research*. 8, 5 (2017), 38.
- [32] Wang, R., Chen, F., Chen, Z., Li, T., Harari, G., Tignor, S., Zhou, X., Ben-Zeev, D. and Campbell, A.T. 2014. StudentLife: Assessing Mental Health, Academic Performance and Behavioral Trends of College Students Using Smartphones. *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (New York, NY, USA, 2014), 3–14.
- [33] Wang, R., Harari, G., Hao, P., Zhou, X. and Campbell, A.T. 2015. SmartGPA: How Smartphones Can Assess and Predict Academic Performance of College Students. *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (New York, NY, USA, 2015), 295–306.
- [34] Wang, Z., Zhu, X., Huang, J., Li, X. and Ji, Y. 2018. Prediction of Academic Achievement Based on Digital Campus. *Proceedings of The 11th International Conference on Educational Data Mining (EDM 2018)* (Buffalo, New York, Jul. 2018).
- [35] Zeng, J., Kruger, U., Geluk, J., Wang, X. and Xie, L. 2014. Detecting abnormal situations using the Kullback–Leibler divergence. *Automatica*. 50, 11 (2014), 2777–2786. DOI:<https://doi.org/10.1016/j.automatica.2014.09.005>.

APPENDIX

Attendance Index Comparison

Let $c_{i,t}$ denoted the index used (raw attendance data $a_{i,t}$ or EMA index $EMA_{i,t}$). In order to illustrate the advantage of EMA over raw data, we predict the attendance rate $a_{i,t}$ using the index $c_{i,t-1}, c_{i,t-2}$ from previous 2 weeks. We construct a model for each week, with the prediction model of each week reflects the properties of the week.

Definition 5 Weekly Prediction Model (WPM): For every week t , the attendance rate of t is predicted as:

$$a_{i,t} = w_{1,t}c_{i,t-1} + w_{2,t}c_{i,t-2} + \varepsilon_{i,t}$$

where $w_{1,t}, w_{2,t}$ are parameters to be estimated.

As we consider attendance rate could be a vibrate behavior influenced by the attendance rates of the previous 2 weeks (two points determine a line), we use the attendance rates of the 2 weeks before time t to predict attendance in week t . The decay factor β for EMA is therefore set to be $\frac{2}{3}$. Model used is linear regression and mean square error (MSE) are calculated for every week (every model) as the evaluation.

Fall terms in both 2018 and 2019 have 14 weeks. Because we used the indexes of previous 2 weeks as predictor, we start to train prediction models beginning from week 3. We then have in total 12 separate models for WPM in a semester.

We use attendance data of students from Cohort 2018 in Fall 2018 semester training set and attendance data of students from Cohort 2019 in Fall 2019 semester as test set.

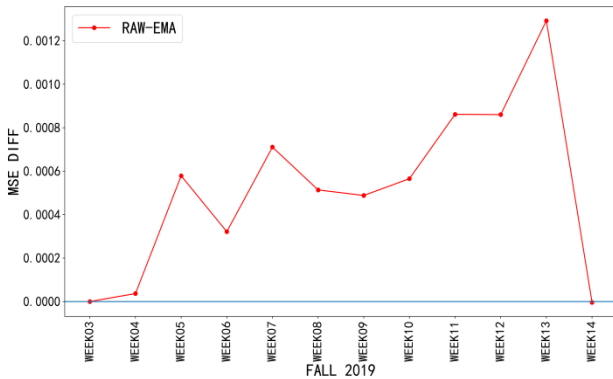


Figure 2. Comparison of test set MSE between raw data and EMA for weekly prediction models.

Figure 2 shows the performance of weekly prediction models for test set data. The difference between original data and EMA (RAW-EMA) is plotted in the figure. A value above zero indicates that RAW has a larger error (worse) than EMA on that point. EMA performs equal to or better than raw data in all 12 models. We can then confirm that EMA can smooth the fluctuating time series and better reveal trend information.

Dissimilarity Selection on Real Data

In this subsection, we compare the performance of KLD over mean difference (MDF) and absolute distance (AD) using real-life attendance data.

To confirm the performance of dissimilarity measurements on real-life data, we apply the three methods on attendance data of

students from Cohort 2018 in week 8 and week 9 in the Fall 2018 semester. We compare the weekly attendance rate between students whose CGPA (measured after Fall 2018) are in the top 25% quantile (297 students) and last 25% quantile (298 students). Weekly attendance rates in a subpopulation are divided into 5 levels, as is illustrated in Figure 3. Larger inner structure difference is detected in week 9 than in week 8, especially for the lowest attendance category (<0.6).

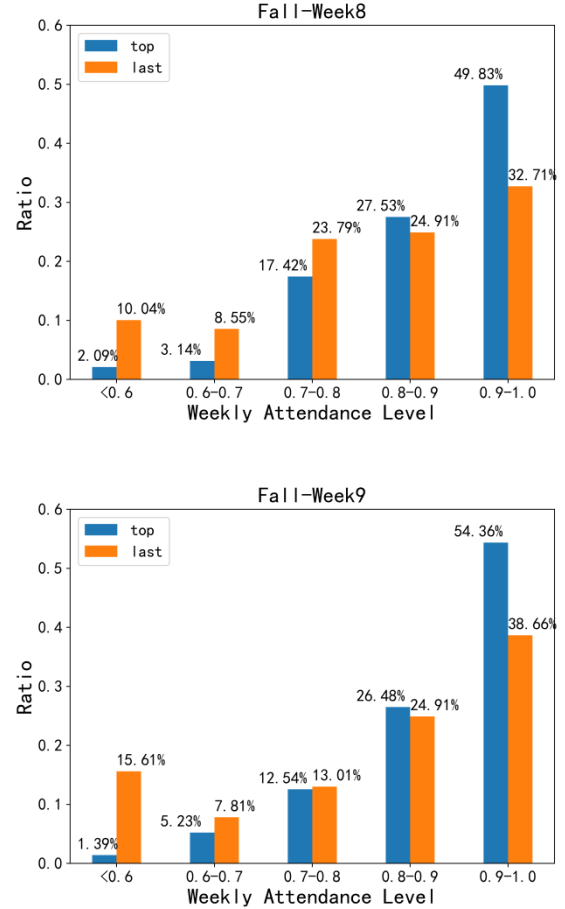


Figure 3. Attendance distribution in Fall 2018 Week 8 & 9

The dissimilarity values between top and last 25% students calculated from the three measurements are shown in Table 3. KLD derives significantly more divergent result for the two weeks (Week 8 and 9), which is consistent with Figure 3. AD cannot detect the divergence order. As KLD could put more emphasis on their most different part, it is more suitable for attendance group comparison.

Table 3. Dissimilarity calculated for the two weeks.

Dissimilarity Measurement	Fall 2018 Week8	Fall 2018 Week9
MDF	0.073	0.083
AD	0.365	0.336
KLD	0.177	0.274

Improved Automated Essay Scoring using Gaussian Multi-Class SMOTE for Dataset Sampling

Jih Soong Tan
Monash University Malaysia
Bandar Sunway
47500 Subang Jaya, Selangor
jih.tan@monash.edu

Ian K.T. Tan
Heriot-Watt University
Malaysia
Precinct 5
62200 Putrajaya
I.Tan@hw.ac.uk

Lay-Ki Soon
Monash University Malaysia
Bandar Sunway
47500 Subang Jaya, Selangor
soon.layki@monash.edu

Huey Fang Ong
Monash University Malaysia
Bandar Sunway
47500 Subang Jaya, Selangor
ong.hueyfang@monash.edu

ABSTRACT

Automated Essay Scoring (AES) research efforts primarily focus on feature engineering and the building of machine learning models to attain higher consensus with human graders. In academic grading such as essay scoring, the scores will naturally result in a normal distribution, more commonly referred to as the bell curve. However, the datasets used do not always have such distribution and are often overlooked in most machine learning environments. This paper proposes a Gaussian Multi-Class Synthetic Minority Over-sampling Technique (GMC-SMOTE) for imbalanced datasets. The proposed GMC-SMOTE generates new synthetic data to complement the existing datasets to produce scores that are in a normal distribution. Using several labeled essay sets, some of which already have a substantial agreement between the machine learning model and human graders, learning from normal distribution datasets yields significant improvements. Improvements of 0.038 QWK score (5.8%) over the imbalanced dataset were observed. The experimental result has also shown that naturally occurring distribution in the automated essay scoring domain contributes to the most appropriate training dataset for machine learning purposes.

Keywords

Boosting, Data Sampling, Gaussian Distribution, Data Pre-Processing, Automated Essay Scoring

1. INTRODUCTION

In an educational setting, essay compositions are commonly used to evaluate students' competence in articulation. The task to grade is labourious and is highly biased to the grader,

J. S. Tan, I. K. T. Tan, L. K. Soon, and H. F. Ong. Improved automated essay scoring using gaussian multi-class SMOTE for dataset sampling. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 647–651, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853024>

which causes the scoring to lack consistency in the essay scoring process [20]. For this reason, Automated Essay Scoring (AES) systems have been proposed and implemented to solve the traditional human scoring approach problem or act as a complementary mechanism. Research projects in AES have focused on feature engineering, and in the design of the scoring machine learning models to achieve higher agreement with human graders [19, 22]. In building the machine learning models, the quality of the dataset is of utmost importance. In the earlier research projects on AES, the distribution of the scoring in the datasets was not taken into consideration. The datasets for building the AES models are often imbalanced, as the scores assigned by human graders may not be appropriately distributed. A significant level of imbalance in multi-class datasets such as essay scoring datasets is a profound problem [21].

1.1 Dataset Distribution

To visualise the imbalanced dataset issue in essay scoring, the score distribution of a commonly used dataset for AES research is used. The dataset is from the Automated Student Assessment Prize (ASAP) competition¹. There are 8 datasets, where sets 1, 2, 7 and 8 are essays of the same genres were chosen and their score distribution is shown in Figure 1, the other sets are not selected as they are of the short letters genre.

Figure 1a and 1b have fewer classes compared to 1c and 1d. The agreement (accuracy) between the model and the ground truth in a multi-class classifier would generally be better with smaller classes. However, for Essay Set 7 (Figure 1c), the model built for it outperforms that of the smaller number of classes Essay Set 2 (Figure 1b) as shown in Table 1b and 1c. Also, it is observed that Figure 1c has a typical academic scoring distribution that has a Gaussian distribution with a median around the 60% mark. It can also be observed that for Figure 1d, the distribution of the scores does not reflect a Gaussian-like distribution with a trough around the 66% mark. As the scoring range for Figure 1d is larger (0 to 60), this would mean that the dataset quality

¹<https://www.kaggle.com/c/asap-aes>

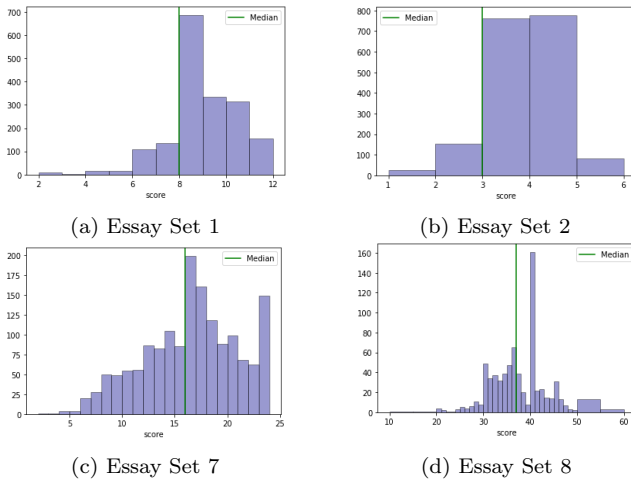


Figure 1: Histogram for ASAP essay sets 1, 2, 7 and 8.

is poor as there are insufficient data that will result in poor performance of the model by comparing the QWK score of Table 1d with 1a, 1b and 1c. Hence, an appropriate dataset distribution is important for training a model to have better performance.

In automated essay scoring, it is well known that the scoring of academic work such as essays normally falls within a Gaussian distribution curve with the median hovering around the 60 - 70 percentile. Motivated by the understanding that quality training dataset will improve the learned model [10, 3], we propose Gaussian distribution Multi-Class Synthetic Minority Oversampling Technique (GMC-SMOTE) to overcome the issues of imbalanced multi-class data for AES modelling. Instead of implementing SMOTE to provide a uniform distribution of the sampling, GMC-SMOTE is used to oversample the multi-class dataset into a Gaussian distribution. With the improved quality of the training set, the inherent bias of the dataset is kept in the training of the models. To evaluate GMC-SMOTE effectiveness, three type of train sets were generated. In addition, the evaluation was conducted using the same features and modelling algorithm, namely the Bayesian Linear Ridge Regression as proposed by Phandi et al. [19].

2. IMBALANCED DATASET HANDLING

2.1 Imbalanced Data Problem Trends

Researchers worked on the imbalanced data problem from two distinctive approaches; modifications on the attributes at the algorithmic level for the training model to fit the imbalanced data, or augmentation at the data level. These two approaches are summarised as follows.

2.1.1 Algorithmic level

Most of the imbalanced data solutions are targeted on the minority class due to the common high cost of misclassifications on a minority class [18]. Hence, numerous cost-sensitive learning approaches are introduced to balance the classes based on the ratio to each classes' costs [17]. The most popular approach of cost-sensitive learning is to assign different weights for each class in the training models that are based on the costs of misclassifications [8]. Other than

assigning weights, cost-sensitive learning can be done differently, such as changing the models' threshold based on its own misclassification costs [8]. However, in most real-world cases, the cost matrix is not easily identified.

2.1.2 Data level

Random undersampling (RU) and oversampling (RO) are the first few methods introduced to deal with imbalanced data. Both of the methods replicate random samples to reduce the imbalance ratio. However, the RU might eliminate some significant samples, and RO might cause the model to overfit. There are several extensions based on random undersampling and oversampling such as one-sided selection [15] and Edited Nearest Neighbour rule (ENN) [25]. Synthetic Minority Oversampling Technique (SMOTE) was introduced by Chawla et al. [6] to address the problem in random oversampling. Instead of randomly replicating the minority samples, SMOTE creates new minority class samples by using the interpolation between minority class samples' neighbourhood. Several extension and hybrid methods based on SMOTE were introduced such as Borderline-SMOTE, SMOTE with Tomek-links and SMOTE with ENN. Eventually, many extensions of SMOTE are proposed such as Borderline-SMOTE [11], ADASYN [13] and MWMOTE [2].

2.2 SMOTE and Multi-Class SMOTE variations

2.2.1 SMOTE

The SMOTE algorithm [6] records the interpolations between minority class instances within a defined neighbourhood to create new synthetic samples. To do so, SMOTE measures the difference between the selected feature vector and its nearest neighbour. SMOTE multiplies the calculated difference by a random number between 0 and 1, then adds it to the selected feature vector. For this reason, the synthetic sample will be at a random point between two specific feature vectors. This method effectively forces the decision making areas for minority class instances to become more generic. However, SMOTE cannot be applied to multi-class problems (such as for AES modeling) directly as SMOTE works on dataset with two classes, the minority, and the majority only. For multi-class problems, there are a few notable multi-class oversampling approaches.

2.2.2 One-versus-All with SMOTE

As multi-class classification implies additional complexity to data mining algorithms, due to the overlapping boundaries, it will lead to a drop in the resulting performance [5]. One of the approaches to tackle this problem is through implementation of class binarization techniques [1]. One-versus-All (OvA), also known as One-versus-Rest (OvR), is a binary classification algorithm for multi-class datasets. The combination of OvA and SMOTE is one of the most popular methods to implement SMOTE for multi-class datasets [10, 3]. OvA is implemented to split the multi-class dataset into multiple binary classification problems. Next, each binary classification problem are trained on a binary classifier based on the samples of selected class as positive and the rest of samples as negative [4]. After obtaining the binary class problems sets from OvA/OvR, SMOTE is implemented for each set to oversample the minority class instances. The

output of SMOTE will be merged back into the multi-class dataset for the training of the selected model.

2.2.3 One-versus-One + SMOTE

One-versus-one (OvO) [12] is another approach of binary classification algorithms for multi-class datasets. Similar to OvA/OvR, OvO binarizes multi-class datasets by splitting a multi-class dataset into binary classification problems [9]. However, OvO deals with the datasets differs in that it splits them into single datasets for each class against every other class.

3. PROPOSED GMC-SMOTE

We propose Gaussian Multi-Class Synthetic Minority Over-sampling Technique (GMC-SMOTE) to enhance the quality of the dataset in AES by introducing new synthetic samples for improved performance of the models. Instead of creating synthetic new samples uniformly, GMC-SMOTE individually processes each class to form a Gaussian-like distribution. The implementation of SMOTE in a multi-class dataset is not directly applicable. To implement SMOTE in a multi-class dataset, we selected OvA with SMOTE to perform the SMOTE for each class in the multi-class dataset as it has the superior performance over oversampling for multi-class dataset [10, 3]. It is one of the techniques suggested for multi-class oversampling by the author of SMOTE [9].

The default SMOTE algorithm requires six samples for each class, which can be reduced to two. In our proposed GMC-SMOTE algorithm, we propose to replicate classes with fewer than six samples to at least six samples. This ensures SMOTE has sufficient samples to generate synthetic samples and inherit the bias from the original dataset. Also, six samples per class are the minimum requirement to train a model [14].

3.1 GMC-SMOTE Algorithm

Algorithm 1 GMC-SMOTE algorithm’s pseudo-code

```

1: Inputs:
2:  $D$  = Dataset classes and its counts
3:  $Dataset$  = Dataset
4: Algorithms:
5:  $C \leftarrow getUniqueClass(D)$ 
6:  $M \leftarrow calculateMode(D)$ 
7: for class in  $C$  do
8:   if  $class = M$  then
9:      $FD_{class} \leftarrow D_M$ 
10:  else
11:     $posAway \leftarrow |M - class|$ 
12:    if  $\frac{D_{class}}{posAway} > 2.5$  then
13:       $FD_{class} \leftarrow D_{class} \times 2.5$ 
14:    else
15:       $FD_{class} \leftarrow \frac{D_{class}}{posAway}$ 
16:    end if
17:  end if
18: end for
19:  $newData \leftarrow SMOTE(Dataset, FD)$ 
20: return  $newData$ 

```

GMC-SMOTE is implemented based on the bell curve symmetric theory [24] for Gaussian distribution. The bell curves symmetric theory means the distribution is symmetric comparing the left distribution and right distribution from the

value at the peak of the curve. Algorithm 1 describes our implementation of the GMC-SMOTE. In line 4-7, C represents unique classes and M represents the mode class in D . Then, each unique class C in D is iterated through. The mode class of the dataset D_M is kept constant. The mode class will be the class with the most occurrences.

$$F_N = \frac{D_{class}}{posAway} \quad (1)$$

The Equation 1 is to calculate the new frequency, F_N for the rest of the classes where $posAway$ represents position away from the mode, and is stored in Frequency Dictionary FD_{class} . This equation is motivated by how the distribution is scaled away from the mode in the bell-curved symmetric. The author of SMOTE, Chawla et al. [6] have proven that between 200% and 300% oversampling rate is proven to be the most robust oversample ratio in SMOTE. Hence, we limit the amount to be oversample at 250% of the original frequency. With this, we can keep the naturally occurring distribution of the datasets and inherit the bias for training the models.

4. METHODOLOGY

4.1 Experimental Datasets

We use the same dataset from Figure 1. The selected datasets metadata can refer to [19]. To extract features for learning algorithms, we implement the Enhanced AI Scoring Engine (EASE) ². We selected EASE as it is a robust feature engineering method for AES that several researchers have implemented in recent years [19, 16]. The EASE system will generate 14 features refer to [19]. Five-fold cross-validation is implemented to generate the train and test set due to unreleased test data from the ASAP competition. We re-distributed the data into five-fold, where four-fold will be the train set and one-fold will be the test set in each round. Three different types of train set will be generated for the learning algorithms to train the models:

- (a) **Default.** The train set with original distribution.
- (b) **Uniform Distribution.** The train set is oversampled by SMOTE that makes the frequencies all classes to be uniformly same as the frequency of the mode class.
- (c) **GMC-SMOTE.** The train set is oversampled by GMC-SMOTE to generate Gaussian-like distribution of the classes.

4.2 Learning Algorithm

The Bayesian Linear Ridge Regression (BLRR) algorithm is chosen among the other prospective methods such as Naive Bayes (NB) and Support Vector Machines (SVM) based on the results from Phandi et al. [19]. It allows a natural language processing tasks to deal with insufficient data by create linear regression through probability distributors instead of point estimate. Also, it is robust and has often delivers good results in natural language processing projects.

²<https://github.com/edx/ease>

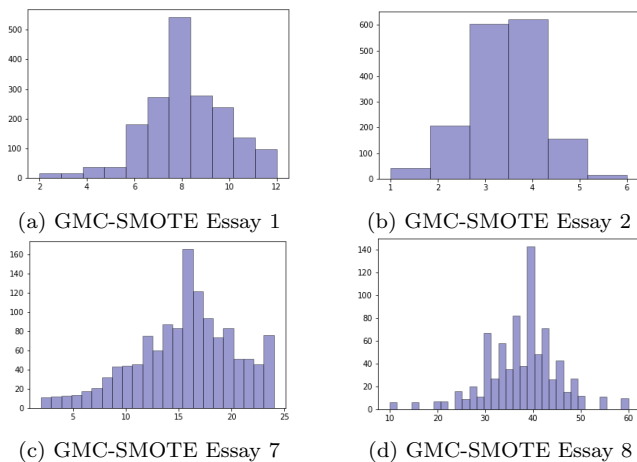


Figure 2: Histogram of GMC-SMOTE generated distributions.

4.3 Evaluation Metric

We implement Quadratic Weighted Kappa (QWK) to calculate the rate of agreement among two graders; the human graders, and the scoring by the trained model. It varies from 0 to 1, where 0 represents no agreement, 0.01-0.20 as slight, 0.21-0.40 as fair, 0.41-0.60 as moderate, 0.61-0.80 as substantial, and 0.81-1.00 as almost perfect agreement [7]. QWK is proven to be robust since it takes into consideration the odds of accidental agreement [23]. Also, it is a common evaluation metric for AES models [19, 16].

5. RESULT AND DISCUSSION

New samples for each essay set using the proposed GMC-SMOTE are generated. A sample of the GMC-SMOTE distribution histograms are plotted in Figure 2. From the observation on the Figure 2a and 2b, the new distributions are closer to a Gaussian distribution symmetrical on the left and right of the mode value. For the Figure 2c, the histogram is much flatter than the default distribution. As for Figure 2d, the data quality remains of concern due to missing data for many classes and low data samples. From all these observations, the nature of default distribution is inherited into the new distribution, which brings the inherent bias of default distribution to the new distribution.

Train Set	QWK
(a)	0.808
(b)	<u>0.810</u>
(c)	0.823

(a) Essay Set 1

Train Set	QWK
(a)	0.688
(b)	0.707
(c)	<u>0.704</u>

(c) Essay Set 7

Train Set	QWK
(a)	0.650
(b)	<u>0.672</u>
(c)	0.688

(b) Essay Set 2

Train Set	QWK
(a)	0.644
(b)	<u>0.656</u>
(c)	0.676

(d) Essay Set 8

Table 1: Experiments result

We calculate the mean QWK scores for each trained model using the one-fold of test in five rounds. The results are

shown in Table 1. The best result is bold-faced and the second best is underlined. Overall, the BLRR scores for GMC-SMOTE datasets are better than all the default datasets and GMC-SMOTE performs the best in essay sets 1, 2, and 8.

- **Essay Set 1:** An increase in QWK by 0.015 even though it has an almost perfect agreement using the default dataset.
- **Essay Set 2:** A more modest increase in QWK by 0.038, which is expected as the dataset distribution is similar and the number of classes are few.
- **Essay Set 7:** The GMC-SMOTE’s QWK is better than the default dataset but slight poorer than Uniform distribution dataset produced by the default SMOTE algorithm.
- **Essay Set 8:** A significant increase in QWK by 0.032 even though it has many missing classes and several classes with one to five frequency counts.

With the exception of essay set 7, the uniformly distributed datasets have poor results. This shows that simply applying SMOTE to the datasets has probably removed the inherent bias in an essay scoring situation. The proposed GMC-SMOTE has shown that with proper oversampling, maintaining the inherent scoring biasness in an academic setting can improve the automatic essay scoring agreement with the human graders. For essay set 7, although the GMC-SMOTE improved the QWK scores over the default dataset, the uniform distributed dataset has the best result. This can be attributed to the fact that the default dataset is relatively platykurtic with negative skewness. This encompasses the majority of the samples (scores of 15 or greater from Figure 1). Hence, applying a uniform distribution will enhance the sample and does not significantly impact the human scoring biasness. It will also mean that if there are sufficient sampling sizes, the naturally occurring distribution is the optimal distribution. This can be observed in Figure 2, for essay set 7, where the histogram is plotted for the new data distribution generated by GMC-SMOTE. Comparing the GMC-SMOTE distribution (Figure 2) with the default distribution (Figure 1), the naturally occurring distribution is kept in the new distribution while new samples are generated.

6. CONCLUSION

The results show that the GMC-SMOTE is an effective oversampling method for situations with some imbalance in the dataset. The proposed GMC-SMOTE method can be applied for training datasets for other classifications domains where the naturally occurring distribution is Gaussian (normal). Kurtosis and skewness can be used to assess the type of naturally occurring distribution of the respective dataset distribution. As observed in our evaluation (essay set 7), it is also important to first assess whether the kurtosis is much lesser than the normal distribution (platykurtic), which may require a uniformly distributed dataset. The assessment can be conducted prior to deciding whether to use Multi-Class SMOTE or the proposed GMC-SMOTE for imbalanced datasets.

7. REFERENCES

- [1] E. L. Allwein, R. E. Schapire, and Y. Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. *Journal of machine learning research*, 1(Dec):113–141, 2000.
- [2] S. Barua, M. M. Islam, X. Yao, and K. Murase. Mwmote—majority weighted minority oversampling technique for imbalanced data set learning. *IEEE Transactions on knowledge and data engineering*, 26(2):405–425, 2012.
- [3] R. C. Bhagat and S. S. Patil. Enhanced smote algorithm for classification of imbalanced big-data using random forest. In *2015 IEEE International Advance Computing Conference (IACC)*, pages 403–408. IEEE, 2015.
- [4] C. M. Bishop. Pattern recognition. *Machine learning*, 128(9), 2006.
- [5] F. Charte, M. J. del Jesus, and A. J. Rivera. *Multilabel classification: problem analysis, metrics and techniques*. Springer, 2016.
- [6] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.
- [7] J. Cohen. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46, 1960.
- [8] C. Elkan. The foundations of cost-sensitive learning. In *International joint conference on artificial intelligence*, volume 17, pages 973–978. Lawrence Erlbaum Associates Ltd, 2001.
- [9] A. Fernández, S. Garcia, F. Herrera, and N. V. Chawla. Smote for learning from imbalanced data: progress and challenges, marking the 15-year anniversary. *Journal of artificial intelligence research*, 61:863–905, 2018.
- [10] A. F. Giraldo-Forero, J. A. Jaramillo-Garzón, J. F. Ruiz-Muñoz, and C. G. Castellanos-Domínguez. Managing imbalanced data sets in multi-label problems: a case study with the smote algorithm. In *Iberoamerican Congress on Pattern Recognition*, pages 334–342. Springer, 2013.
- [11] H. Han, W.-Y. Wang, and B.-H. Mao. Borderline-smote: a new over-sampling method in imbalanced data sets learning. In *International conference on intelligent computing*, pages 878–887. Springer, 2005.
- [12] T. Hastie, R. Tibshirani, et al. Classification by pairwise coupling. *Annals of statistics*, 26(2):451–471, 1998.
- [13] H. He, Y. Bai, E. A. Garcia, and S. Li. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*, pages 1322–1328. IEEE, 2008.
- [14] V. Indira, R. Vasanthakumari, and V. Sugumaran. Minimum sample size determination of vibration signals in machine learning approach to fault diagnosis using power analysis. *Expert Systems with Applications*, 37(12):8650–8658, 2010.
- [15] M. Kubat, S. Matwin, et al. Addressing the curse of imbalanced training sets: one-sided selection. In *Icml*, volume 97, pages 179–186. Citeseer, 1997.
- [16] S. Latifi and M. Gierl. Automated scoring of junior and senior high essays using coh-matrix features: Implications for large-scale language testing. *Language Testing*, 38(1):62–85, 2021.
- [17] X.-Y. Liu and Z.-H. Zhou. The influence of class imbalance on cost-sensitive learning: An empirical study. In *Sixth International Conference on Data Mining (ICDM’06)*, pages 970–974. IEEE, 2006.
- [18] D. Margineantu. When does imbalanced data require more than cost-sensitive learning. In *Proceedings of the AAAI’2000 Workshop on Learning from Imbalanced Data Sets*, pages 47–50, 2000.
- [19] P. Phandi, K. M. A. Chai, and H. T. Ng. Flexible domain adaptation for automated essay scoring using correlated linear regression. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 431–439, 2015.
- [20] M. D. Shermis and J. C. Burstein. *Automated essay scoring: A cross-disciplinary perspective*. Routledge, 2003.
- [21] M. A. Tahir, J. Kittler, and A. Bouridane. Multilabel classification using heterogeneous ensemble of multi-label classifiers. *Pattern Recognition Letters*, 33(5):513–523, 2012.
- [22] J. S. Tan and I. K. Tan. Feature group importance for automated essay scoring. In *International Conference on Multi-disciplinary Trends in Artificial Intelligence*, pages 58–70. Springer, 2021.
- [23] S. Vanbelle and A. Albert. A note on the linearly weighted kappa coefficient for ordinal scales. *Statistical Methodology*, 6(2):157–163, 2009.
- [24] T. Wang, H. Li, Z. Li, and Z. Wang. A fast parameter estimation of generalized gaussian distribution. In *2006 8th international Conference on Signal Processing*, volume 1. IEEE, 2006.
- [25] D. L. Wilson. Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on Systems, Man, and Cybernetics*, (3):408–421, 1972.

A Topic-Centric Crowdsourced Assisted Biomedical Literature Review Framework for Academics

Ryan Hodgson
Durham University
ryan.t.hodgson
@durham.ac.uk

Jingyun Wang
Durham University
jingyun.wang
@durham.ac.uk

Alexandra I. Cristea
Durham University
alexandra.i.cristea
@durham.ac.uk

Fumiko Matsuzaki
Kyushu University
fumu@bioreg.kyushu-
u.ac.jp

Hiroyuki Kubota
Kyushu University
kubota@bioreg.kyushu-
u.ac.jp

ABSTRACT

In the academic process, comprehension and analysis of literature is essential, however, time-consuming. Reviewers may encounter difficulties in identifying relevant literature, given the considerable volume of available texts. It is arduous not only for starting PhD students, but also for any researcher learning about a new field (called here "domain learners"). To address this issue, we present an *automated framework to assist in the literature review process*. Through the application of topic modelling of academic articles, our framework encourages senior researchers within a specific field to act as experts to contribute to the labelling of topics. Further to this, domain learners can benefit from visualisation tools intended to assist in the comprehension of vast amounts of academic texts. Our approach allows reviewers to identify the *topics, trends* as well as *relations between topics* in a given research field. We also accompany this method with a tool that we provide open source. For illustration, we apply here our method to a case-study of biological texts, specifically texts related to human protein kinases. To further enhance the educational capabilities of our approach, we perform triangulation of external biomedical databases, to illustrate how our multi-pronged approach can provide a comprehensive understanding of the research domain.

Keywords

Automated Literature Review, Topic Modelling, Crowdsourcing, Bioinformatics

1. INTRODUCTION

The process of understanding academic literature is a time-consuming process for both students and professionals. Thus, there is a necessity to enable the process of quickly comprehending such literature. Furthermore, limitations to the

amount of work which can be analysed by an individual or team may be encountered, due to the time it takes to understand each item of literature. This is of particular relevancy to the biomedical field, where estimates in 2016 proposed that the field observes 3 new publications per minute uploaded to the PubMed database alone [1]. Moreover, the identification and recognition of evidence and named entities within full-text biological literature - the Curator Assistance Problem [1], is limited by the requirement for manual analysis of text by experts within the field, using their own knowledge and intuition.

To address these issues, we propose an *automated approach to assist in the literature review process in biomedical literature*. By leveraging recent advancements in topic models, we seek to provide novel benefit to the academic and research process when learning about a certain new research area. Further to this, to address the Curator Assistance Problem [1], we perform rule-based identification of human-related protein kinases within the literature, for automatic triangulation of multiple external data sources, to assist in comprehension of research. We provide access to the case-study for reproducibility¹. Our research aims to address the following research questions: 1. *How may topic modelling algorithms be applied to assist in the comprehension of large volumes of academic data?* 2. *How may semantic similarity measures be leveraged to identify inter-topic relationships of identified literature?* 3. *Can rule-based extraction of named biomedical resources contribute to the comprehension of automatically generated literature topics?* 4. *Can scientific language-oriented embedding models provide improvements to the perceived accuracy of topic-modelling, when analysed by biological domain-experts?*

The main contributions of this paper are thus: (1) Provision of a novel framework for the analysis of literature topics by accounting for semantic relationships and temporal trends of identified topics. (2) A capability for the crowdsourcing [2] of domain experts for the labelling and filtering of topics within the literature. Experts such as educational tutors or senior researchers may contribute to the labeling of identified topics based upon their own knowledge. Subsequently, domain learners may then benefit from the visualisation tools

R. Hodgson, J. Wang, A. Cristea, F. Matsuzaki, and H. Kubota. A topic-centric crowdsourced assisted biomedical literature review framework for academics. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 652–656, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853002>

¹https://github.com/ryanon4/topic_labelling_tool

provided to assist in their learning on the subject. (3) Triangulation of multiple external data sources for the extraction and linking of named kinases present within literature, to assist in the expert labelling and comprehension of topics within the literature. Although this is a case-study based upon literature related to protein kinases, in a collaboration with bioscientists, this robust approach allows for further generalisation, and could be applied to the analysis of literature from any domain. (4) An unsupervised evaluation of the performance of scientific-domain transformer embeddings, compared to document embedding approaches when applied to document clustering for the identification of topics within scientific literature.

2. RELATED WORKS

As a data-intensive field, there has been an emergence of numerous biological data-sources of various quality and provenance. However, due to a lack of a standard ontology with fine granularity, it is difficult to conduct data mining on those heterogeneous biology data. Despite efforts to normalise data formats and collect observations in databases, a large amount of information relevant to biology research is still recorded as free text in journal articles and in comment fields of databases [3]. Therefore, retrieving information from papers and merging it with existing biological databases can provide a crucial foundation for data-mining in bioinformatics [1]. To address this, works have been proposed for the extraction of entities within biological texts, including RegulonDb [4], which applied rule-based identification of regulatory interactions in *Escherichia coli*, with the results demonstrating a rule-based approach sufficient in identifying 45% of all named entities when compared with a manual extraction approach. Extraction of genetic and protein interactions was performed by BioC-BioGRID, which released a corpus of 120 full-text articles with biological and molecular entity annotations [5]. Based upon these problems and subsequent approaches, [1] proposed the Curator Assistance Problem. This problem proposes four objectives, which were defined given a full-text biological research paper: 1. The recognition of evidence described in an article, compared to information in other articles. 2. The recognition of evidence which is supported by experimental data, compared to hypothetical or vague statements. 3. The distinction between statements on layout, compared to statements of results. 4. The recognition of negative statements.

No research was identified relative to the automated literature review process for biomedical data. However, in the wider domain, one framework has been demonstrated to identify relevant papers for a literature review based upon an input of seed papers [6]. This approach applied supervised learning classifiers, which were trained upon reference lists of existing papers. From an unsupervised perspective, [7] applied topic modelling to the task of conducting an exploratory literature review through the use of Latent Dirichlet Allocation (LDA). The approach required application of the elbow curve methodology within an exhaustive search, to determine the optimum number of topics within the literature. In contrast, a recent topic modelling approach, Top2Vec [8], has presented advantages over the application of LDA, through the elimination of the need to manually or exhaustively define the optimal topic number. Applications of Top2Vec to assist in literature analysis were presented in

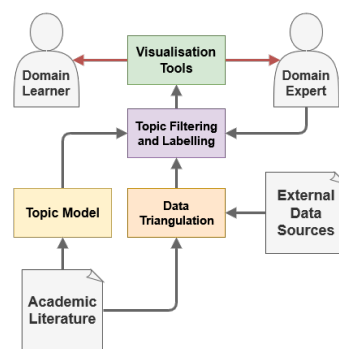


Figure 1: An automated framework based on topic modelling and crowdsourcing

[9], through a case-study review of research in Intelligent Tutoring Systems. However, the approach required the manual analysis of resulting topics prior to the generation of visualisation and topic labels. Our study is based on this, but is expanding it in several ways. Importantly, we propose a framework for the e-learning domain, and illustrate it through an interface, to permit the crowdsourcing [2] of topic labelling on a large scale, as well as provide an expanded suite of tools to assist with learning. By creation of this framework, we are able to demonstrate the generalisable characteristics of the approach, such that it may be applied to any domain of literature, and may be subsequently enhanced through the implementation of domain-relevant features, such as the protein databases applied for our case-study. The resulting implementation of domain-relevant data triangulation may enhance the information available to researchers by providing supplementary information to aid in filtering and identification of relevant literature to a domain-user.

3. METHODOLOGY

The proposed framework (as summarised in Figure 1) entails three components. Firstly, topic modelling is performed on the academic literature that is identified as relevant to the case-study domain through the use of the API resource. Following this, senior researchers are presented with an interactive topic labelling and visualisation analysis tool, which allows them to determine whether a topic is relevant to their literature search, as well as assign a title for that topic. Finally, this visualisation analysis tool can facilitate the comprehension of relevant literature of the domain learners. To illustrate the framework, here, based on the full-text of documents assigned to a topic, extraction of human-protein-kinase terms (the topic of the literature review) is performed by rule-based extraction of kinase names. Also, in terms of the actual implementation, an internal relational database provides external links to the external resources – here, those coming from UniProt Knowledgebase (UniProtKB) [10] and InterPro [11], well-known protein databases. The resulting interface may be applied as a framework for learning through the crowdsourcing of domain experts within a field for the labelling and filtering of literature topics. Expertly annotated topics may then be provided to learners or experts, who may apply the visualisation tools provided to assist in their learning and comprehension of the literature.

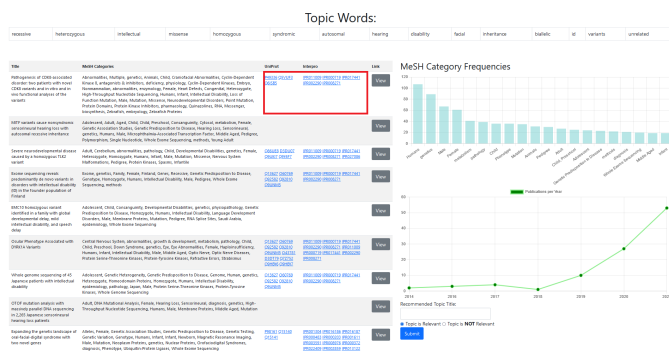


Figure 2: Labelling Interface for Topic "Genetic Disorders"

3.1 Data Sources

A full-text representation of publications is provided through the PubMed API, which ensures to capture the full semantic context of a text if it is analysed by a compatible language model for variable length document-embeddings. Two additional sources were selected for the triangulation of kinases identified within the literature, with these being the UniProtKB [10] and Interpro [11] databases. UniProtKB is a database of protein sequences and functional information, which includes information relevant to specific proteins, their function, organism, presence in biological processes, and relevant literature. InterPro serves as a similar platform, however includes hierarchical family classifications of protein entries as well as domains and functional sites found in proteins.

For the case-study, a list of 624 human protein kinases was firstly obtained from a kinase database [12, 13]. A comprehensive list of human proteins with links to their respective InterPro IDs was obtained from UniProtKB (Swiss-Prot, Jan 2014 version) [10]. Matching of protein kinase names to gene names including synonyms for proteins from UniProtKB was performed. The resulting 357 protein kinases that successfully matched to UniProtKB entries were used in this study. This entailed the names of relevant genes, kinase names and kinase families. Further to specific kinase names (e.g. *AATK1*), the data included full-named entities relevant to a kinase. Relevant to the given example, this would be *Apoptosis-associated tyrosine kinase 1*. This list of shortened and full terms was applied in an automated literature search via the PubMed API [14]. A total of 19,258 records were returned in this manner. Following this, filtering was performed, to ensure the elimination of any noise introduced into the dataset during the search stage. This was performed via filtering out publications that failed to contain Medical Subject Heading (MeSH) terms related to Humans. Following elimination of non-human-related subjects, 14,631 documents remained.

3.2 Topic Modelling

For the identification of topics within our literature, the Top2Vec [8] algorithm, which leverages the HDBSCAN [15] algorithm, is applied to identify dense clusters of document embeddings, and determine these to be topics. The algorithm automatically generates human-readable labels, based upon the top-scoring topic-words for each topic. In addition, a domain-expert may manually assign a title for each

topic on the topic labelling screen. The use of HDBSCAN presents a unique opportunity when clustering, given that HDBSCAN may label documents determined to be noise [15]. This provides the benefit of eliminating documents which do not directly fall within a topic.

3.3 Literature Analysis Toolkit

The main contribution of our approach centers around the development of a simple user interface for evaluating the results of literature topic analysis, and subsequent deeper analysis. Researchers may find difficulty in the application of algorithmic analysis approaches, as these require a degree of knowledge of programming and an understanding of the underlying algorithms applied. Therefore, through providing an interactive user interface, we ensure that researchers from the biomedical domain may analyse literature without the need to spend time learning the underlying system. Results from the topic modelling of the literature are presented to the domain-expert in a simple landing page, which provides a table format of top-scoring words for each topic, the manually assigned label, and relevancy classification which can be manually defined. By clicking on the *Edit Label* button, domain-experts may view a detail page (as shown in Figure 2), featuring top-scoring publications assigned to a topic, the top-scoring words for the topic, and a bar graph visualisation, detailing the total frequency of MeSH terms for publications assigned to that topic. Providing these features presents domain-expert with an opportunity to decide the title of the topic through a number of means. For example, MeSH subject headings can provide the frequency in which certain diseases are mentioned within documents in a topic. Further to this, by extracting mentions of human protein kinases within full-text articles, we facilitate the triangulation of the InterPro [11] and UniProtKB [10] databases, any may provide domain-expert with external links (as shown in the highlighted box in Figure 2) to related resources relative to a given publication. Domain-expert may also view the full publication at the place of publishing through a hyperlink, so that individual publications they find useful through the above analysis methods may then be read further.

Further academic features are provided to users (both domain experts and learners) through an analysis of the semantic relationships within topics, following the same methodology presented by [9]. Semantic relationships are generated through the calculation of the cosine similarity between all combinations of topic vectors, where a topic vector is defined as the average of all document embeddings belonging to that topic. We define each topic as a node, with the top three highest scoring topic similarity pairs (edges) for each topic being presented in the graph visualisation. Nodes are presented with at least three edge relationships. However, some nodes may have more than three edges. For instance, as shown in Figure 3, the node "DNA repair" displays the top 3 scoring edge relationships with "Genetic disorders," "Cancer genetics" and "Phosphorylation profiling". However, For the "Radiation effects" and "Alternative splicing" nodes, the "DNA repair" edge relationship is within the top-3 ranking for the nodes respectively, hence "DNA repair" demonstrating 5 edge relationships.

The topic-relationship graph is computed when initialising the system, and store this in a serialised format. Edge thick-

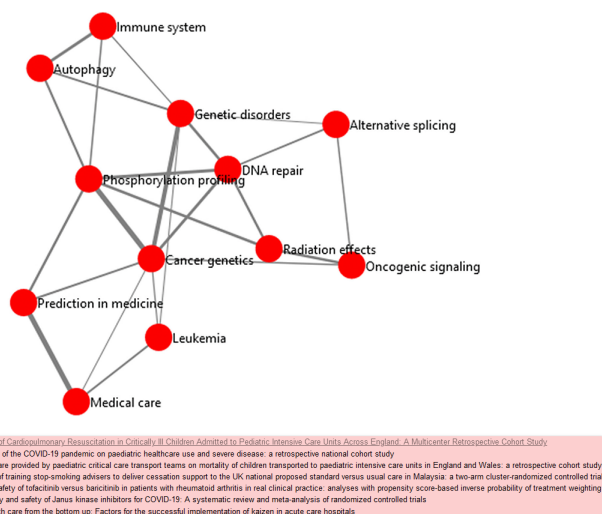


Figure 3: Relationships of Topics Identified by the Bioscientist Team and Expanded Link View

ness is a reflection of the cosine-similarity score between two nodes, for easy understanding of the strength of the relationship between those topics, with a tooltip presenting the similarity score when the user hovers their mouse over an edge. From this view, users may also further expand a topic, navigating to a view of the top-scoring academic texts that have been assigned to that topic based upon the distance of the publication to the topic centroid, by right-clicking on a node. For example, the pink panel on the bottom of Figure 3 is displayed when right-clicking the node "Medical care". This panel provides links of the paper directly to the external resources. Users may choose to view all identified topic-relationships from the topic modelling process, or only those that have been manually labelled as relevant.

4. CASE STUDY IN BIOLOGY - HUMAN PROTEIN KINASES

As a case study, the toolkit generated 279 topics from texts of publications related to the 357 human protein kinases. A representative list of associations, for each topic, is provided and contains the top twenty highest scoring topic-words, top ten individual publications, and MeSH category frequencies of publications assigned to the topic. For example, the 10th topic (shown in Figure 2) is provided with the following information: relevant topic words such as "recessive," "heterozygous," "inheritance," "missense," "homozygous," "syndromic," "autosomal," "disability," and "variants"; titles of individual publications with phrases such as "neurodevelopmental disease caused by a homozygous TLK2 variant," "variants in disorders with intellectual disability," and "phenotype associated with DYRK1A variants"; and MeSH categories such as "Humans" and "Genetics" with the highest frequency in the assigned publications. Based on this information, the topic can be labeled as "Genetic disorders" associated with human protein kinases. Moreover, links to UniProtKB and InterPro databases are listed for specific kinases included in each individual publication. For example, the publication titled "Pathogenesis of CDK8-associated disorder: two patients with novel CDK8 variants and in vitro

and in vivo functional analyses of the variants" is provided with the following links: the entry of the kinase CDK8 in UniProtKB; and the entries of domains, binding-sites, etc. found in CDK8 such as "Protein kinase domain," "Protein kinase, ATP binding site," and "Serine/Threonine protein kinases active-site" in InterPro. Such external databases provide detailed information such as biological functions, disease association, sequences, and domain architectures of protein kinases along with links to further explore other databases, should such a need arise for the user.

The toolkit also easily visualises relationships among topics based on their semantic similarity. To demonstrate its efficacy, the bio-experts in our team selected 12 topics that have at least six out of ten individual publications with links to protein kinases in UniProtKB, and then manually labeled the topics as shown in the previous paragraph. The resulting graph network (shown in Figure 3) generated from these topics has 12 nodes with the assigned labels and edges with different thickness that reflects the similarity between the linked topics. The node "Genetic disorders" is strongly connected to the nodes of "Cancer genetics" and "DNA repair." This is consistent with the idea that both genetic disorders and cancer genetics involve genetic changes and genomic technologies. Similarly, DNA repair defects are associated with certain genetic disorders. For example, ATM and ATR are DNA repair-associated serine/threonine kinases and their genetic mutations can cause hereditary disorders, ataxia telangiectasia and Seckel syndrome, respectively. Thus, it is reasonable to include both ATM and ATR in topic-words for the topic "DNA repair." Another strong connection visualised by the network is between "Medical care" and "Prediction in medicine," which aligns with the clinical aspect of treatment. The visualisation enables the users to easily explore major topics and their relationships. Collectively, the toolkit allows users to navigate a new field through vast amounts of literature in an efficient way.

5. CONCLUSIONS

In this work, an automated literature review framework is proposed for the analysis of large volumes of literature. The resulting topics from topic modelling of academic literature within specific domain, are provided via an interactive UI tool, which may allow the manual analysis and filtering of resulting topics, and exploration of entailing literature. Based on a case study on the biomedical domain of human protein kinases, we provide a further contribution to the comprehension of literature through the triangulation of biological relations present in identified publications. From an educational data mining perspective, our tool achieves the goal of grouping large amounts of academic text into understandable topics, and allows for crowdsourcing of expert-knowledge for the labelling of topics identified in the literature. The resulting framework achieves two objectives by serving as a base literature review assistance tool, or as an e-learning utility to allow expert analysis of topics, before providing the results to learners. By the inclusion of external databases UniProtKB and InterPro, we provide assistance in the extraction of named kinases within individual items of literature, allowing users to easily read further into the identified kinases present. Overall, this framework incorporates topic modelling with a crowd-sourcing approach, achieving the goal of expediting the task of analysing large volumes of literature.

6. REFERENCES

- [1] Jacques Nicolas. *Artificial Intelligence and Bioinformatics*, pages 209–264. Springer International Publishing, Cham, 2020.
- [2] Larissa Hammon and Hajo Hippner. Crowdsourcing. *Business & Information systems engineering*, 4(3):163–166, 2012.
- [3] Lynette Hirschman, Jong C Park, Junichi Tsujii, Limsoon Wong, and Cathy H Wu. Accomplishments and challenges in literature data mining for biology. *Bioinformatics*, 18(12):1553–1561, 2002.
- [4] Carlos Rodríguez-Penagos, Heladia Salgado, Irma Martínez-Flores, and Julio Collado-Vides. Automatic reconstruction of a bacterial regulatory network using natural language processing. *BMC bioinformatics*, 8(1):1–11, 2007.
- [5] Rezarta Islamaj Doğan, Sun Kim, Andrew Chatr-aryamontri, Christie S. Chang, Rose Oughtred, Jennifer Rust, W. John Wilbur, Donald C. Comeau, Kara Dolinski, and Mike Tyers. The BioC-BioGRID corpus: full text articles annotated for curation of protein–protein and genetic interactions. *Database*, 2017, 01 2017. baw147.
- [6] Jason Portenoy and Jevin D. West. Constructing and evaluating automated literature review systems. *Scientometrics*, 125(3):3233–3251, Dec 2020.
- [7] Claus Boye Asmussen and Charles Møller. Smart literature review: a practical topic modelling approach to exploratory literature review. *Journal of Big Data*, 6(1):93, Oct 2019.
- [8] Dimo Angelov. Top2vec: Distributed representations of topics. *CoRR*, abs/2008.09470, 2020.
- [9] Ryan Hodgson, Alexandra Cristea, Lei Shi, and John Graham. Wide-scale automatic analysis of 20 years of its research. In Alexandra I. Cristea and Christos Troussas, editors, *Intelligent Tutoring Systems*, pages 8–21, Cham, 2021. Springer International Publishing.
- [10] The UniProt Consortium. UniProt: the universal protein knowledgebase in 2021. *Nucleic Acids Research*, 49(D1):D480–D489, 11 2020.
- [11] Matthias Blum, Hsin-Yu Chang, Sara Chuguransky, Tiago Grego, Swaathi Kandasamy, Alex Mitchell, Gift Nuka, Typhaine Paysan-Lafosse, Matloob Qureshi, Shriya Raj, Lorna Richardson, Gustavo A Salazar, Lowri Williams, Peer Bork, Alan Bridge, Julian Gough, Daniel H Haft, Ivica Letunic, Aron Marchler-Bauer, Huaiyu Mi, Darren A Natale, Marco Necci, Christine A Orengo, Arun P Pandurangan, Catherine Rivoire, Christian J A Sigrist, Ian Sillitoe, Narmada Thanki, Paul D Thomas, Silvio C E Tosatto, Cathy H Wu, Alex Bateman, and Robert D Finn. The InterPro protein families and domains database: 20 years on. *Nucleic Acids Research*, 49(D1):D344–D354, 11 2020.
- [12] Gerard Manning, David B. Whyte, Raquel Martinez, Tony Hunter, and Sucha Sudarsanam. The protein kinase complement of the human genome. *Science*, 298:1912 – 1934, 2002.
- [13] Protein kinases, kinomes and evolution, at kinase.com.
- [14] Entrez programming utilities help, Jan 1970.
- [15] Ricardo J. G. B. Campello, Davoud Moulavi, and Joerg Sander. Density-based clustering based on hierarchical density estimates. In Jian Pei, Vincent S. Tseng, Longbing Cao, Hiroshi Motoda, and Guandong Xu, editors, *Advances in Knowledge Discovery and Data Mining*, pages 160–172, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

Personalized and Explainable Course Recommendations for Students at Risk of Dropping out

Kerstin Wagner

Berliner Hochschule für Technik, Germany
kerstin.wagner@bht-berlin.de

Agathe Merceron

Berliner Hochschule für Technik, Germany
merceron@bht-berlin.de

Petra Sauer

Berliner Hochschule für Technik, Germany
sauer@bht-berlin.de

Niels Pinkwart

Humboldt-Universität zu Berlin, Germany
niels.pinkwart@hu-berlin.de

ABSTRACT

This paper presents a course recommender system designed to support students who are struggling in their first semesters of university and who are at risk of dropping out. Considering the needs expressed by our students, we recommend a set of courses that have been passed by the majority of their nearest neighbors who have successfully graduated. We describe this recommender system, which is based on the explainable k -Nearest Neighbors algorithm, and evaluate the 2nd and 3rd semester recommendations using historical data. The evaluation reveals that the recommendations correspond to the actual courses passed by students who graduated, whereas the recommendations and actually passed courses differ for students who dropped out. The recommendations show to struggling students a different, ambitious, but hopefully feasible way through the study program. Furthermore, a dropout prediction confirms that students are less likely to drop out when they pass the courses recommended to them.

Keywords

Course recommender system, nearest neighbors, explainability, user-centered design, dropout prediction.

1. INTRODUCTION

In the last decades, universities worldwide have changed a lot. They offer a wider range of degree programs and courses and welcome more students from diverse cultural backgrounds. Further, teaching and learning in high schools differ from teaching and learning in universities. Some students cope well and keep the same level of academic performance at university as they had in high school. Others struggle, perform worse, and might become at risk of dropping out. The preliminary exploration of our data has shown, that most of the students drop out during the first three semesters of their studies. Therefore, the course recommendations proposed in this work focused on supporting struggling students after their 1st and 2nd semester. The final goal in developing such a system is to have it integrated in novel facilities that universities could put in place to support their diverse students better.

K. Wagner, A. Merceron, P. Sauer, and N. Pinkwart. Personalized and explainable course recommendations for students at risk of dropping out. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 657–661, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853008>

At the beginning of each semester in Germany, students need to decide which courses they enroll in. When entering university directly after high school for their 1st semester, most of them decide to enroll in exactly the courses planned in the study handbook. The decision becomes more difficult when they fail courses in their 1st semester and should choose the courses to enroll in their 2nd semester: should they repeat right away the courses they failed? Which courses planned in the 2nd semester in the study book should they take? Should they reduce the number of courses they enroll in to have a better chance of passing them all? Should they take more courses to compensate for the courses they failed? The study handbook does not help in finding answers to those questions.

In our previous work [18], most of the students mentioned that they rely on friends and acquaintances as one source of information when deciding which courses to enroll in. Further, any system assisting in enrollment should have explainable recommendations.

In this work, we propose a recommendation system to support students choosing which courses to take before the semester begins. We recommend students the set of courses that the majority of their nearest neighbors, who successfully graduated, have passed. Nearest neighbors are students who, at the same stage in their studies, have failed or passed almost the same courses with the same or very similar grades. The system proposed in the present work does not recommend top N courses as other systems do, e.g. [10, 14]. Rather, it recommends an optimal set of courses, and we assume that a student should be able to pass all the courses of that set. Because the recommendations are driven by past records of students who graduated, we also pose the hypothesis that students following the recommended set should have a lower risk of dropping out.

We describe a recommender system based on the explainable algorithm k -Nearest Neighbors (k -NN) and evaluate the recommendations given after the 1st and the 2nd semester using historical data. Although the recommendations are designed to support struggling students, every student should have access to them. By contrast, the recommendations should show a different, more academically successful way of studying to struggling students and therefore differ from the courses that they pass. More precisely, this paper tackles the following research questions:

RQ1. How large is the intersection between the set of recommended courses and the set of courses a student has passed? Are there differences between struggling and well-performing students?

RQ2. Do the recommendations lower the risk of dropping out, and if so, how much?

2. RELATED WORK

Course recommendations. Supporting students in choosing courses at the start of the semester has been studied in many works. The aims of the proposed recommender systems are diverse. For example, Parameswaran et al. [13] sought to provide students with courses that meet their constraints like availability and schedule, but also being favored and chosen by other students. Their goal was to recommend interesting courses that will also help students graduate and conduct evaluations using the academic performance data of students who have graduated. The goal of Pardos and Jiang [14] was to recommend courses “that are novel or unexpected to the student but still relevant to their interests”. The authors recommended courses based on a chosen favored course and evaluated the results using both historical data and student feedback as part of a user study. Backenköhler et al. [1] optimized recommendations by combining three aspects: a student’s preparedness for a course, the benefit of a course towards other courses, and the predicted performance in the course. The authors used historical data to evaluate how the recommendations match the student’s actual course choices. Nearer to our aim, Elbadrawy and Karypis [4] and Morsy and Karypis [10] recommended courses for which students can expect a good grade or an increase in their GPA. In both papers, the authors evaluated recommendations based on historical data, as we do.

User-centered design. Our work follows a user-centered approach as has been proposed for example by Martinez-Maldonado et al. [9]. The design of our recommendations has been developed with respect to insights obtained from a semi-structured group discussion conducted with 25 students [18]. The authors of several studies described their user-centered approach to involve stakeholders in the development of tools. De Quincey et al. [16] included students in the development of a dashboard that integrates study motivation to track engagement and predicted scores at Keele University (UK). To inform the development of a dashboard with relevant indicators to help students choose what courses to take in an upcoming academic period, Hilliger et al. [7] identified student information needs regarding course enrollment at Pontifical Catholic University (Chile) using a mixed-methods approach. Sarmiento et al. [17] described their approach as a series of co-design workshops for learning analytics tools with and for students at New York University.

Explainability. Not all machine learning algorithms used in educational data mining are explainable, such as neural networks, which are increasingly used as stated by Barredo Arritea et al. [2]. Pardos and Jiang as well as Morsy and Karypis used them in their respective works, [10, 14] for instance. According to Ning et al. [11], neighbor-based approaches are simple, justifiable, efficient, and stable. The number of neighbors, the features, and the distance function all influence the explainability of k -NN [2]. The nearest neighbors can be visualized as a table and students understand why the recommendations are given to them. As argued by Williamson and Kizilcec in [19], “educators and learners will not trust a model that cannot easily be explained to them”. Indeed, it was a concern of our students: recommendations should be explainable [18].

Our contribution. Using student-centered design, we have developed an approach to course recommendations that is explainable to students and aims to help students who are at risk of dropping out. Our evaluation, based on historical data, distinguishes two groups: students who have dropped out and students who have graduated. First, we compare the recommendations to the set of courses that students passed; this is similar to other evaluations like [4, 10].

Second, we investigate whether our recommendations decrease the dropout-risk. This is a novelty of our work.

3. DATA AND METHODOLOGY

Data and preprocessing. For the development of the course recommendation system, data from a six-semester bachelor program of a medium-sized German university was used. The initial dataset contains 1,484 students who started their study program between winter semester 2012 and summer semester 2019. We removed three types of students: A) outliers regarding the number of passed courses based on the interquartile range (IQR) since students can receive credit for courses completed in previous study programs and thus may pass more courses in a semester than foreseen in the study handbook [12], B) students who were still studying at the time of data collection since they can not be used to predict the risk of dropping out, and C) students without at least one record in each of the first three semesters. The final dataset contains 578 students who either graduated (status G) or dropped out (status D) and 9,500 records. The grading scale for passing a course is from best to worst [1.0, 1.3, 1.7, 2.0, 2.3, 2.7, 3.0, 3.3, 3.7, 4.0]. The grade for failing an exam is 5.0. Students must pass all mandatory courses and four elective courses to graduate. The study handbook includes a suggested courses schedule for the six semesters and students may follow this schedule or not. At any time in their studies, they are allowed to choose courses from all offered courses. It is worth noting that electives courses are scheduled in the 5th and 6th semester of the program. Table 1 shows the final number of students by student status and per semester the number of different courses, the number of academic performance records, the average number of courses passed per student, and the average grade. 134 students have the status dropout (D) and 444 the status graduate (G); in the 3rd semester (bottom line) there were 686 records concerning 20 different courses for students with the status D and 2,622 records concerning 26 courses for students with the status G. One notices that students with the status D pass fewer courses per semester and receive lower grades.

Table 1. Dataset overview by semester S (1, 2, 3) and student status (D, G). Number of courses C, Number or records R, mean number of passed courses MPC, and mean grade MG.

			C		R		MPC		MG	
D	G	S	D	G	D	G	D	G	D	G
		1	14	16	657	2,199	3.6	4.7	2.8	2.1
134	444	2	15	19	740	2,596	3.4	5.1	3.0	2.3
		3	20	26	686	2,622	3.0	5.4	3.2	2.2

Data representation. We use only data about academic performance: each student is represented by a vector of grades. It is possible for a student to, for example, enroll in a course in the 1st semester and not take the exam, then enroll and fail the exam in the next semester and enroll again and pass the exam in the following semester. In this case, a student has three different records for the same course in three different semesters. In our opinion, not only the final grade with which a course was passed is relevant, so we include the entire history of a student’s academic performance in the vector.

Missing values. To compute the nearest neighbors, we have to deal with missing values. If a student was enrolled in a course but did not take the exam, a 6.0 was imputed, if s/he was not enrolled at all, a 7.0 was imputed. This means that we rate enrolling but not taking the exam (6.0) more similar to failing (5.0) than not enrolling (7.0).

5. COURSE RECOMMENDATIONS

For our course recommendation system, we use the idea of a k -NN classifier: given a student S_x represented by the vector x_n of length n at the end of semester t , we use the majority votes of his/her already graduated neighbors to obtain a set of courses for the following semester $t+1$ that are classified as “passed” and accordingly recommended; any course not in that set is not recommended. We recommend courses for the 578 students in the dataset and then evaluate the recommendations.

Procedure. Let Y be the data of a program after semester t consisting only of graduated students S_{GR} , i.e. their vectors of grades with length n , let S_x be a student who needs a course recommendation, let C be the set of all courses of the program and let k be the number of nearest neighbors. Given C , the expected output is C_R the set of recommended courses and all courses that are in C but not in C_R are not recommended. First, we determine the k nearest neighbors of an observed student: the similarity of students is calculated using the Euclidean Distance between x and y where S_x and S_y are two students and x and y are the vectors of all their grades at the end of semester t where the length of the vectors n corresponds to the number of features that we use. The students are sorted by increasing distance and the top k vectors are selected as the neighborhood S_k of student S_x . Instead of considering each course in C , we preselect only courses that at least one of the students in S_k has passed in semester $t+1$, i.e. the grade is lower or equal to 4.0, and assign them to the set C_P of courses passed by the neighborhood. To classify each course in C_P to be recommended or not, we use the majority vote of the k neighbors: if a neighbor passed a course, it is labeled with 1 and otherwise 0. We calculate the probability P for student S_x to pass that course c as the mean of class labels, e.g. if $k=3$ and 2 out of 3 neighbors have passed the course, P is 0.66. If the mean is higher than 0.5, the course is recommended and assigned to C_R . To avoid a tie in majority voting, we use only uneven k and test our approach with $k = 1, 3, 5, 9, 19$. We check if S_x has already passed a recommended course and remove this course from the recommendation if necessary.

Baseline. We use the same approach not only with neighbors who graduated but also with all neighbors, i.e. students who dropped out and students who graduated. We expect that the recommendations differ between the two approaches and that the recommendations based on graduated students, but not necessarily the recommendations generated with all students, reduce the risk of dropping out. In the following, we distinguish the two recommendation types with AN (all neighbors) and GN (graduated neighbors).

6. COURSES’ INTERSECTION (RQ1)

To compare the recommended courses with those actually passed, we consider the recommendations as a binary classification problem and build a confusion matrix to evaluate the recommendation for each student and each semester as follows: a course recommended and actually passed in semester $t+1$ will be a true positive (TP), a course recommended and actually not passed will be a false positive (FP), a course not recommended but passed will be a false negative (FN), and a course not recommended and not passed will be a true negative (TN). The choice of metric is critical due to the relatively large number of courses and the resulting imbalance of recommendations or non-recommendations. Further, because we recommend a set of courses and not top N courses, it is crucial to measure not only that the recommendations contain all passed courses (recall) but also that they do not contain courses that students did not pass (precision). We chose the F_1 score to evaluate courses’ intersections since the F_1 score ignores true negatives,

which is in our context always a high value, and thus serves our needs. The score ranges from 0 to 1 with 1 indicating perfect classification (recall=1 and precision=1) and 0 indicating perfect misclassification (recall=0 or precision=0). The calculation is as follows: $F_1 = 2 \cdot TP / (2 \cdot TP + FP + FN)$.

Results. To evaluate the quality of the recommendation, we look at the intersection and evaluate the recommendations by F_1 score. We expect a high score for students with the status G in both semesters as the recommendations for them should match closely the courses that they passed. We do not expect a high score for students with the status D; the recommendations are meant to show them another way of studying that should bring more academic success and therefore should not match closely the courses that they passed. Table 2 provides the results as mean F_1 scores of all students. To better distinguish for which student groups the recommendations are more appropriate, the results are grouped by the following factors: semester (2, 3), student status (D/G), and type of neighbors (AN/GN). In addition, we provide the difference in each row between the mean score of the baseline AN and the mean score of GN. The differences in the number of neighbors k are not shown because no large differences emerged when k varied.

Findings. There is a difference between students who struggle and students who perform well. More precisely: A) AN-based and GN-based recommendations: The mean F_1 score for dropouts and graduates is higher for GN than for AN. B) Semester: The mean F_1 score is higher after the 1st for the 2nd semester than after the 2nd for 3rd semester. C) Graduates and dropouts: The mean F_1 score is higher for graduates than for dropouts. The mean F_1 score is high for graduates: this confirms our expectation that recommendations closely match the courses passed by graduates. For students with status D, the mean F_1 score tends to be low: the recommendations show another way of studying.

Table 2. Courses’ intersection based on mean F_1 score of all students for semester S and recommendation types AN and GN as well as their differences (Diff) by student status (D, G).

S	D			G		
	AN	GN	Diff	AN	GN	Diff
2	0.471	0.509	0.038	0.843	0.861	0.017
3	0.282	0.302	0.020	0.809	0.836	0.027

7. CHANGES IN DROPOUT RISK (RQ2)

A dropout prediction was performed using the following two steps: 1) based on the actual enrollment and exam information, a model has been trained to predict the two classes dropout or graduate; 2) the model of step 1 is used again to predict dropout or graduate after replacing the actual enrollment and exam information by the calculated recommendations. We call dropout risk the proportion of students who are predicted “dropout” by a model. To determine whether or not the recommendation approach helps to lower the dropout risk, we analyze the difference of these two dropout predictions.

Feature set. As investigated by Manrique et al. [8], there are several ways to select a feature set for dropout prediction and no way works better than the others in all contexts. Because we want to measure the impact of our recommendations on dropout prediction, and the individual courses are relevant accordingly, we use the courses taken by students as features and the grades obtained by students as their values.

Model training. To detect a change in the dropout risk in step 2, the models should be as accurate as possible which we aimed to

achieve through two approaches: A) train different types of algorithms including hyperparameter optimization (Logistic Regression, LASSO Regression, Decision Tree, k -Nearest Neighbors, Support Vector Classifier with different kernels (radial basis function, linear, polynomial), Random Forest), and B) use algorithm-independent parameters for optimization since we realized that hyperparameter optimization alone was insufficient. B1) Feature selection: we removed features with a low number of actual grades and tried different thresholds as a minimum number of actual grades for a course: (1, 5, 9, 19). B2) Training data balancing: we used two common techniques: Synthetic Minority Over-sampling Technique (SMOTE) [3] and RandomOverSampler (ROS). Both implementations are from the Python library `imbalanced-learn`. B3) Decision threshold moving: Usually, a classifier decides for the positive class at a probability greater or equal to 0.5, but in case of imbalanced data, it may be helpful to adjust this threshold, so we checked additionally to 0.5 the values: 0.1, 0.3, 0.7, 0.9.

Models selection. To emphasize that both correct dropouts and correct graduates are important for dropout risk prediction, we evaluated our models based on the test data with *Balanced Accuracy* (BACC) as the mean of the recall for class 1 (dropout) and recall for class 0 (graduate). The data sets sorted by the start of study were split into 80% training data and 20% test data, so that risk prediction is done for students who started their studies last. We trained models for both semesters $t=2$ and $t=3$ with actual grades and used the best models to evaluate a change in dropout risk in step 2. In both cases, RF achieved the highest BACC for step 1 (2nd semester: 0.859, 3rd semester: 0.935). The algorithm-independent parameters belonging to the models are: features selection (2nd: 0, i.e. no features were removed, 3rd: 1), decision threshold (2nd: 0.3, 3rd: 0.5), and balancing the training data (2nd: SMOTE, 3rd: ROS). Regarding hyperparameter optimization, we did not observe any improvements in BACC, which we relate to the small size of the training sets.

Step 2 dropout prediction. We used the selected models to predict dropout based on the recommendations. Since we assume the student will pass the recommended courses, we need a grade between 1.0 and 4.0 for step 2. If we had a grade in the records for that student and a recommended course, we used it. If the student had dropped the course or failed it, we imputed the average of two medians: the median of all the grades that the student has earned so far and the median of the historical grades for that course. We evaluated this imputation with the data we use in this work and obtained an *Root Mean Square Error* (RMSE) of 0.634, which is comparable with the RMSE from 0.63 to 0.73 in [4, 15].

Results. Table 3 shows three proportions of students who are predicted as dropout (P1, P2_AN, P2_GN) and the differences between these proportions. P1 is the prediction based on actual enrollment and exam data. P2_AN corresponds to step 2; the dropout prediction uses the courses recommendations calculated with all neighbors while P2_GN uses the recommendations calculated with graduate neighbors. The differences in the number of neighbors k are not considered because no large differences emerged when k varied: P2_AN and P2_GN are the average values from the risk predictions, based on the test data set. The three columns on the right provide the differences in the predictions: P2_AN vs P1, P2_GN vs P1, P2_GN vs P2_AN. For example, 81.4% of the actual dropout students for semester 2 are predicted as dropout in the first prediction, 75.8% using AN-based recommendations, and 70.2% using GN-based recommendations. The dropout risk based on the GN recommendation is 5.6% lower than the prediction based on the AN recommendation.

Findings. It turns out that, with our strong assumption that students will pass the recommended courses, the risk of dropping out can be reduced. More precisely: The proportion of students predicted as dropout are lower when the predictions are AN- and GN-based in step 2 than using the actual enrollment and exam data in step 1. The GN-based recommendations provide a lower dropout risk compared to the AN-based recommendations. The proportion of students predicted as dropout are lower for the 3rd than for the 2nd semester using AN- and GN-based recommendation. The dropout risk reduction in step 2 is higher for dropouts than for graduates.

Table 3. Mean predicted risk in step 2 dropout prediction (P2) for recommendation types AN and GN compared to step 1 (P1) by student status ST (D, G) and semester (S).

ST	S	P1	P2_AN	P2_GN	P2_AN vs P1	P2_GN vs P1	P2_GN vs P2_AN
D	2	0.814	0.758	0.702	-0.056	-0.112	-0.056
	3	0.884	0.665	0.293	-0.219	-0.591	-0.372
G	2	0.096	0.074	0.060	-0.022	-0.036	-0.014
	3	0.014	0.047	0.005	0.033	-0.008	-0.041

8. DISCUSSION AND CONCLUSION

In this paper, we have presented an explainable course recommender system designed primarily to support students who are struggling after their 1st or 2nd semester at university. The recommendations are based on the explainable k -NN algorithm and are built by selecting the courses that most of the nearest neighbors who graduated have passed. We have evaluated our approach on historical data in two ways. First, we have compared the recommendations with the set of the courses that students have passed using the F_1 score. Second, we have investigated whether students are less likely of dropping out when following the recommendations. Further, we also have evaluated the impact of choosing nearest neighbors from the set of students who dropped out and graduated, our baseline, instead of choosing them only from the set of students who graduated.

The F_1 score evaluating the recommended courses is higher when the neighbors are chosen from the set of students who graduated, as can be seen in Table 2. It is particularly high, mainly over 80%, for students with the status graduate, which confirms that, for them, the recommendations match closely the courses that they pass. Consistent with this finding, the number of students who are predicted with the status dropout is smaller when the recommendations are used in the prediction rather than the actual data. Preliminary work shows that these findings generalize to other degree programs.

The results suggest that the provided recommendations would help more students to graduate if the recommendations are both ambitious and realistic: students indeed do pass the courses recommended to them. A closer look at the recommendations reveals that a small number of students receive an empty set, which should be examined in detail. Further, it still needs evaluations with students in their 1st or 2nd semester on how ready and willing they are to use such recommendations, and which extra support they need to pass all courses recommended to them. As stated by Falk et al. [6] in the German context, most of the time a combination of well-orchestrated interventions brings academic success.

In terms of k -NN, it would be worth testing multilabel learning as shown by Zhang and Zhou [20] and if the same approach could be used for planning over several semesters, as proposed by Erzhuo et al. [5]. Finally, we would like to investigate which other approaches are equally visualizable and explainable to students.

9. REFERENCES

- [1] Backenköhler, M., Scherzinger, F., Singla, A., and Wolf, V. 2018. Data-Driven Approach towards a Personalized Curriculum. In *Proceedings of the 11th International Conference on Educational Data Mining*, 246–251.
- [2] Barredo Arrieta, A., Díaz-Rodríguez, N., Del Ser, J., Benetot, A., Tabik, S., Barbado, A., Garcia, S., Gil-Lopez, S., Molina, D., Benjamins, R., Chatila, R., and Herrera, F. 2020. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion* 58, 82–115.
- [3] Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. 2002. SMOTE: Synthetic Minority Over-sampling Technique. *JAIR* 16, 321–357.
- [4] Elbadrawy, A. and Karypis, G. 2016. Domain-Aware Grade Prediction and Top-n Course Recommendation. In *Proceedings of the 10th ACM RecSys Conference on Recommender Systems*. Association for Computing Machinery, 183–190. DOI=10.1145/2959100.2959133.
- [5] Erzhuo Shao, Shiyuan Guo, and Zachary A. Pardos. 2021. Degree Planning with PLAN-BERT: Multi-Semester Recommendation Using Future Courses of Interest. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence. AAAI Special Track on AI for Social Impact*. 35, 17, 14920–14929.
- [6] Falk, S., Tretter, M., and Vrdoljak, T. 2018. Angebote an Hochschulen zur Steigerung des Studienerfolgs: Ziele, Adressaten und Best Practice. *IHF kompakt*, March 2018.
- [7] Hilliger, I., Laet, T. de, Henríquez, V., Guerra, J., Ortiz-Rojas, M., Zuñiga, M. Á., Baier, J., and Pérez-Sanagustín, M. 2020. For Learners, with Learners: Identifying Indicators for an Academic Advising Dashboard for Students. In *Proceedings of the 15th European Conference on Technology Enhanced Learning. Addressing Global Challenges and Quality Education*. Lecture Notes in Computer Science. Springer International Publishing, Cham, 117–130. DOI=10.1007/978-3-030-57717-9_9.
- [8] Manrique, R., Nunes, B. P., Marino, O., Casanova, M. A., and Nurmikko-Fuller, T. 2019. An Analysis of Student Representation, Representative Features and Classification Algorithms to Predict Degree Dropout. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge*. Association for Computing Machinery, New York, NY, USA, 401–410. DOI=10.1145/3303772.3303800.
- [9] Martinez-Maldonado, R., Pardo, A., Mirriahi, N., Yacef, K., Kay, J., and Clayphan, A. 2016. LATUX: an Iterative Workflow for Designing, Validating and Deploying Learning Analytics Visualisations. *Learning Analytics* 2, 3, 9–39.
- [10] Morsy, S. and Karypis, G. 2019. Will this Course Increase or Decrease Your GPA? Towards Grade-aware Course Recommendation. *Journal of Educational Data Mining* 11, 2, 20–46.
- [11] Ning, X., Desrosiers, C., and Karypis, G. 2015. A Comprehensive Survey of Neighborhood-Based Recommendation Methods. In *Recommender Systems Handbook*, F. Ricci, L. Rokach and B. Shapira, Eds. Springer US, Boston, MA, 37–76. DOI=10.1007/978-1-4899-7637-6_2.
- [12] Novoseltseva, D., Wagner, K., Merceron, A., Sauer, P., Jessel, N., and Sedes, F. 2021. Investigating the Impact of Outliers on Dropout Prediction in Higher Education. In *Proceedings of DELFI Workshops 2021*. DELFI 2021 - 19. Fachtagung Bildungstechnologien der GI. Hochschule Ruhr West, 120–129.
- [13] Parameswaran, A., Venetis, P., and Garcia-Molina, H. 2011. Recommendation systems with complex constraints. *ACM Transactions on Information Systems* 29, 4, 1–33.
- [14] Pardos, Z. A. and Jiang, W. 2020. Designing for serendipity in a university course recommendation system. In *Proceedings of the 10th International Conference on Learning Analytics & Knowledge*. Association for Computing Machinery, New York, NY, USA, 350–359. DOI=10.1145/3375462.3375524.
- [15] Polyzou, A. and Karypis, G. 2016. Grade Prediction with Course and Student Specific Models. In *Proceedings of the 20th Pacific Asia Conference on Knowledge Discovery and Data Mining*, 89–101. DOI=10.1007/978-3-319-31753-3_8.
- [16] Quincey, E. de, Briggs, C., Kyriacou, T., and Waller, R. 2019. Student Centred Design of a Learning Analytics System. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge*. Association for Computing Machinery, New York, NY, USA, 353–362. DOI=10.1145/3303772.3303793.
- [17] Sarmiento, J., Campos, F., and Wise, A. 2020. Engaging Students as Co-Designers of Learning Analytics. In *Companion Proceedings 10th International Conference on Learning Analytics & Knowledge*, 29–32.
- [18] Wagner, K., Hilliger, I., Merceron, A., and Sauer, P. 2021. Eliciting Students’ Needs and Concerns about a Novel Course Enrollment Support System. In *Companion Proceedings of the 11th International Conference on Learning Analytics & Knowledge*, Online, 294–304.
- [19] Williamson, K. and Kizilcec, R. F. 2021. Effects of Algorithmic Transparency in Bayesian Knowledge Tracing on Trust and Perceived Accuracy. In *Proceedings of the 14th International Conference on Educational Data Mining*, 338–344.
- [20] Zhang, M.-L. and Zhou, Z.-H. 2007. ML-KNN: A lazy learning approach to multi-label learning. *Pattern Recognition* 40, 7, 2038–2048.

A deep reinforcement learning approach to automatic formative feedback

Aubrey Condor
University of California Berkeley
aubrey_condor@berkeley.edu

Zachary Pardos
University of California Berkeley
pardos@berkeley.edu

ABSTRACT

Receiving formative feedback about open-ended responses can facilitate the progression of learning. However, educators cannot often provide immediate feedback and thus for students, learning may be slowed. In this paper, we will explore how an automatic grading model can be coupled with deep Reinforcement Learning (RL) to create a system of automatic formative feedback on students' open-ended responses. We use batch (offline) learning with a double Deep Q Network (DQN) to simulate a learning environment, such as an open-source, online tutoring system, where students are prompted to answer open-ended questions. An auto-grader is used to provide a rating of the student's response, and until the response is scored at the highest category, an RL agent iteratively provides suggestions to the student to revise the previous version of their answer. The automated suggestion can include either a key idea to consider adding to the response, or a recommendation to delete a specific part of the response. Our experiments are based on a simulated environment, within which we anticipate a how a real student might revise their answer based on the agent's chosen hint. Preliminary results show that in such environment, the agent is able to learn the best suggestions to provide a student in order to improve the student's response in the least number of revisions.

Keywords

Formative Feedback, Deep Reinforcement Learning, Automatic Short Answer Grading

1. INTRODUCTION

It has been shown that the use of open-ended (OE) items is beneficial for student learning by self-explanation [2], or information recall [1]. Additionally, receiving formative feedback – information communicated to the learner intended to modify his or her thinking/behavior to improve learning – about OE responses may also contribute to the progression of learning for students [16]. However, assessing OE items

and subsequently providing formative feedback is time consuming for teachers [6] and consequently, feedback to students can be delayed.

We suggest the use of educational technology to help mitigate this issue. Technologies like online learning platforms or intelligent tutoring systems can enable students to take a proactive role in assessing their own abilities. In addition, they can help promote equitable learning for students who do not have the resources to receive quick and individualized feedback from a human tutor, parent, or private educator. Such platforms have the capability to provide instant feedback, allowing for students, of all types, to take control of their own learning through real-time formative self-assessment [11]. However, items that elicit instant feedback necessitate a structure for automatic grading, and the creation of useful feedback. While implementing automatic grading and producing related feedback for multiple choice items is seamless, as a student's response is limited to a small number of choices, it is not so simple for OE questions that elicit an infinite number of potential student responses.

In this paper, we explore how an Automatic Short Answer Grading (ASAG) model can be coupled with deep Reinforcement Learning (RL) to create a system that provides automatic formative feedback for students' OE responses. We will use batch (offline) learning to simulate an environment, such as an open source tutoring system, where students are prompted to answer OE questions. Once the ASAG model provides a rating of the response, the RL agent will give a hint to the user by suggesting either a key phrase that the user might consider adding to the response, or a portion of the response that the user might want to delete when revising their answer. The student's revision will be simulated by either the key phrase being added to the previous response at the agent's chosen index within the response, or the portion to consider removing deleted completely. The new response will be once again sent to the auto grader to be classified. This process will continue until the automated rating has reached some defined threshold (as a function of the reward), or a maximum number of hints has been provided.

The key idea of our work is to train an RL agent to choose the best sequence of revisions for an OE response, in order to arrive at an exemplary response in the least number of revisions. Although the space of student responses to a given question is infinite, we hypothesize that the agent can

A. Condor and Z. Pardos. A deep reinforcement learning approach to automatic formative feedback. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 662–666, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853061>

learn which of a finite set of revisions will be most useful to improve a student’s answer.

The main contribution of this work is to explore a real-world application of deep RL. There has been little work so far studying the use of deep RL in educational contexts, and in addition, coupling deep RL with language models (the auto grader) has not been extensively investigated. In this paper, we will briefly describe the ASAG, but will mainly focus on the implementation of the deep RL algorithm.

2. RELATED WORK

Several works have focused on the use of RL for applications in education. Reddy et. al (2017) created a model-free review scheduling algorithm to learn a policy that operates on raw observations of a student’s study history, as opposed to explicitly modeling the student as employed by other scheduling algorithms [13]. Iglesias et. al (2009) used RL within a pedagogical module of an education system such that the system would automatically learn which pedagogical policy was best for a particular student [7]. Dorça et. al (2013) presented an automatic probabilistic approach for modeling student’s learning styles based on reinforcement learning [5]. They show that because of the dynamic aspects of detecting learning styles, the RL agent is able to constantly adjust to a student’s performance. Park et. al (2019) introduced an RL “social robot” for personalized and adaptive education [12]. They illustrate how the agent can utilize children’s verbal and nonverbal affective cues to exert influence over the student’s engagement, in order to maximize long-term learning gains. Rowe et. al (2015) employ modular RL, a multi-goal extension of classic RL, to dynamically tailor narrative-centered learning to particular students’ needs [14]. They show that including a data-driven, planning approach can enhance student learning. Finally, Shawky & Badawi (2018) use RL to build an intelligent environment that provides a method to account for the continuously-changing states of student learners [15].

3. BACKGROUND

In this section we will provide a brief overview of the methods used in our work necessary to understand the results, including the ASAG model, offline (batch) learning, Q-learning and Epsilon-greedy sampling.

3.1 Deep Reinforcement Learning

Reinforcement learning (RL) employs learning to control a dynamical system by providing feedback to an agent, via a reward system, in order to choose an action to take in a given state. The action consists of what the agent can do in the state, where the state represents the current condition of the agent. When the agent takes an action, it receives a reward (either positive or negative) as feedback from the environment. Thus, the agent learns a policy to maximize its total expected sum of rewards. The system can be defined as a fully or partially-observed Markov decision process (MDP) [17].

Deep RL utilizes high-capacity function approximators - deep neural networks - as the policy, in conjunction with RL. The incorporation of deep networks has improved performance for applications of RL in many domains [8]. With deep RL,

the agent can make decisions from unstructured data without any manual engineering of the state space and the algorithms can handle very large amounts of data for learning an optimal policy.

3.2 The ASAG model

The Automatic Short Answer Grading (ASAG) model that we use to provide an initial rating of student responses, as well as feedback to the agent through a reward function about the chosen action (response revision), is a BERT-base multi-class classification model. The BERT transformer language model, introduced in [4], is pre-trained on large amounts of natural language text from Wikipedia and BooksCorpus, and can be fine-tuned for downstream tasks such as classification. We use a compressed version of the model called BERT-base, and fine-tune the model as a supervised classifier using human ratings of the OE question(s) as ground-truth ratings.

3.3 Offline (batch) learning

Traditionally, RL algorithms employ online learning by iteratively collecting experiences, i.e. data, while actively interacting with the environment. The collected experiences are used by the model to improve the policy [17]. For some applications, however, using online data collection can be resource heavy or dangerous or impractical for settings that necessitate a large dataset [8]. In contrast, offline RL algorithms utilize previously collected data. Data is stored in a replay buffer and is not altered during training, allowing for the incorporation of very large, or pre-existing datasets. Issues posed by fully offline-learning algorithms include, most prominently, a vulnerability to distributional shift.

3.4 Deep Q Learning

Q-learning is categorized as a model-free reinforcement learning algorithm - i.e. it does not use the transition probability distribution or the reward function associated with the MDP. Essentially, the model-free algorithm is based on trial-and-error. For any finite MDP, the Q-learning algorithm will find an optimal policy that maximizes the expected total reward over all steps in the sequence, starting from the current state [9]. The Q-function determines the value of a state-action pair through the given reward. Deep Q-learning is a variant of Q-learning where a nonlinear function approximator - a neural network - represents Q.

4. TRAINING THE RL AGENT

In this section, we describe the dataset used for initial results as well as the RL formulation consisting of the state, reward, and action set.

4.1 The Dataset

To evaluate our proposed methods, we take the most simple approach and use only one OE science question from an open-source data set called the Automatic Student Assessment Prize, Short Answer Scoring data (ASAP-SAS). The data was used in a 2012 Kaggle competition¹ sponsored by the Hewlett Foundation, and consists of almost 13,000 short answer responses to 10 science and English questions. The questions were scored from 0 (most incorrect) to 3 (most

¹<https://www.kaggle.com/c/asap-sas>

correct), and each question includes a scoring rubric. We chose the question that achieved the highest validation accuracy, evaluated with a Cohen’s Kappa Metric, with the automatic grading model. We did so such that the reward signal, as it is a function of the autograder, would be strong and consistent, rather than using a question that the autograder has a difficult time scoring and may subsequently provide inconsistent reward feedback to the agent.

The question prompt asks students to *“List and describe three processes used by cells to control the movement of substances across the cell membrane.”* And the rubric states that an answer rated at the highest level will include three correct processes, down to the lowest level which will include zero correct processes. Examples of key statements that can/should be included in an answer, according to the rubric, include: *“Selective permeability is used by the cell membrane to allow certain substances to move across”,* or *“Osmosis is the diffusion of water across the cell membrane”*. The list of key statements include in the rubric drove the action space selection of phrases that the RL agent can add to a response (to simulate suggesting that a student consider the information in their revision).

4.2 The Algorithm

An overview of the proposed algorithm is as follows. For each episode, we will first initialize by randomly sampling one of the student’s short answers. The student answer text will be vectorized and fed to the agent. Based on this state input, the agent will choose an action to take, i.e., either adding a key phrase or deleting a portion of the response. This simulates the student taking the hint into account when revising their answer. Next, we send the new student response to our language model classifier, and receive a probability distribution of rating categories as output. Then we will calculate our reward as a function of the difference in class probabilities from the previous response, to the newly revised response, such that an increased probability of the highest rating category for the revised response would correspond to a higher reward. If the response has reached a high rating, the episode will terminate. Otherwise, the process is repeated with the revised student response (the original response plus the hint phrase, or minus the deleted portion) as the new state, and the agent will choose another action until the highest rating is achieved, or we reach a pre-defined maximum number of revisions. Details of the process are described below.

4.2.1 The State

The state is primarily represented by a student’s response. As our RL agent will not take in words as input, we vectorize the student responses using a Word2Vec [10] model, with embeddings of size 8, and a vocabulary specific to our dataset. Word2Vec produces an embedding for each word in the vocabulary, so in order to create one embedding for the entire response, we concatenate the individual word embeddings. W2V embeddings of size 8 are particularly small, but we chose to sacrifice some information within the word embeddings in order to keep our state space at a reasonable size. Additionally, we concatenate each word embedding instead of averaging them, which is a common method for producing sentence embeddings because we want the agent

to have a capacity to interpret the response at the word level, and not only as an aggregate response.

4.2.2 The Reward

The reward is a function of the difference in the ASAG model’s classification probability distribution of ratings between the old response and the revised response, with a penalty for each revision the agent makes. We include the penalty (subtracting 1) because we want the agent to learn how to revise the student’s response in the least number of revision steps. Additionally, we more heavily weighted the higher rating category probability changes because we care more about a change in the highest rating probabilities, as it is the ultimate goal to achieve the highest level rating. Finally, we multiply the weighted difference in probabilities by three, because the overall difference tended to be small, and we needed to send a stronger signal to the agent. The Reward formula is as such, where Δp_0 represents the difference in probability of the 0 rating category between the new and old response:

$$3 \cdot (0 \cdot \Delta p_0 + 1 \cdot \Delta p_1 + 2 \cdot \Delta p_2 + 3 \cdot \Delta p_3) - 1$$

4.2.3 The Action Space

The action space consists of both adding key phrases to the response or deleting a portion of the response. The agent can choose only one revision (one addition, or one deletion) in each step. The key phrases the agent can add must be pre-defined by either a subject-matter-expert or created based on a detailed scoring rubric, in order to determine what is necessary for a student to include in their explanation to achieve a high rating. For the question in our initial experiment, *“List and describe three processes used by cells to control the movement of substances across the cell membrane.”*, the list of 10 key phrases that the agent can choose from are shown below.

[‘energy to move’], [‘across cell membrane’], [‘diffusion substance across’], [‘active transport requires’], [‘osmosis water across’], [‘passive transport requires’], [‘no energy move’], [‘from high concentration’], [‘to low concentration’], [‘with concentration gradient’]

The phrases are vectorized with Word2Vec the same way the response is for the state space. The phrase can be added to the response at any index in increments of 3 (i.e., the phrase can be added to the very beginning of the response at index 0, or three words in at index 3, etc.). We hypothesized that it is necessary that the agent chooses where, within the response, the phrase will be added. Additionally, the action space includes removing any one trigram within the response. We limited the flexibility of the agent removing text to trigrams to keep the action space at a reasonable size. With a larger action space, it is reasonable to assume that the agent will take much longer to learn.

4.2.4 The Q Network

The Q-Network consists of a fully connected neural network with two hidden layers of size 300 and 200. The final layer corresponds to the dimension of the action space. In addition, both hidden layers use an Rectified Linear Unit (ReLU) activation function. As input, the network takes in the state space (vectorized student response) and outputs an estimate

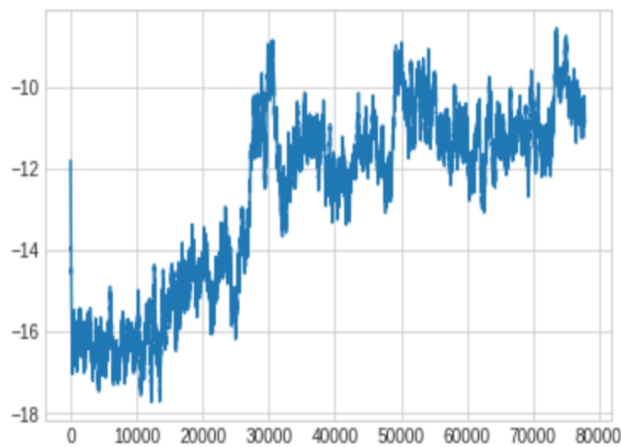


Figure 1: Sum of rewards over 80K training episodes.

of the expected future sum of rewards of each action in the action space. Thus, the agent chooses the action that has the maximum Q Network output. Our algorithm is optimized using Adam stochastic optimization [3]. The size of the replay buffer for the Q-Network is 10,000. In addition, the Q-Network is trained with a batch size of 32, a discount factor of 0.99, and a learning rate of $5e-4$. We update the target network every 4 episodes.

5. RESULTS

Preliminary results shown in Figure 1 imply that the agent is indeed able to learn which revisions to make to a response in order to achieve a higher rating from the autograding model. Figure 1 shows the sum of rewards in an episode on the y-axis, and the episode number along the x-axis. The scale or number of the y-axis is not necessarily meaningful, as our reward function is only created as a means to give signal to the agent about the goodness of its sequence of decisions. Rather, the overall pattern of the reward becoming less negative, i.e. greater, as the training episodes increase shows us that the agent can learn, over many episodes of trial and error, an optimal policy to choose which actions to take, in order to maximize its future sum of rewards. It is worth noting that with limited computational resources, training the agent over 80,000 episodes is not quick, so future work will indeed include training over a greater number of episodes, to see if the agent can achieve a somewhat convergence to a higher total episodic reward than is shown in Figure 1.

6. DISCUSSION

We emphasize that this project is a work in progress and acknowledge that there is much more effort needed to explore the agent’s capacity to revise a student’s response. We believe that it is an important and successful first step to observe that the RL agent is indeed able to learn the task that we proposed, as there exist many sequential tasks that may not be appropriate for the RL formulation, or are too complicated for an RL agent to learn. Using a reward that is a function of a language model poses a unique challenge as we adopt the limitations of the autograding model itself

into our reward formulation. Additionally, using concatenated word vectors as a state space adds an additional layer of complexity because the variability in the state space is infinite, as any student may use an infinite combination of words in their response.

More importantly, we note that there are several practical application of an RL agent revising a student’s response. Our results are from a simulated environment that is meant to represent a real-life student interaction, but does not do so perfectly. When thinking forward to the real-life application of implementing the algorithm in the context of a real student revising their answer, many unanswered questions arise. Firstly, we assume that a student will take into account the RL agent’s revision suggestion directly in our simulation (the agent automatically adds the key phrase or deletes the segment), but a real student will have the choice to consider the suggestion or ignore it and make a different revision. This may pose confusing signals to the agent about whether or not the suggestion was the best one. Further, we must investigate how exactly to provide the hints, as formative feedback, such that the student actively learns from their revision.

7. REFERENCES

- [1] S. Bertsch, B. J. Pesta, R. Wiscott, and M. A. McDaniel. The generation effect: A meta-analytic review. *Memory & cognition*, 35(2):201–210, 2007.
- [2] M. T. Chi, N. De Leeuw, M.-H. Chiu, and C. LaVancher. Eliciting self-explanations improves understanding. *Cognitive science*, 18(3):439–477, 1994.
- [3] K. Da. A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [4] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [5] F. A. Dorça, L. V. Lima, M. A. Fernandes, and C. R. Lopes. Comparing strategies for modeling students learning styles through reinforcement learning in adaptive and intelligent educational systems: An experimental analysis. *Expert Systems with Applications*, 40(6):2092–2101, 2013.
- [6] C. L. Hancock. Implementing the assessment standards for school mathematics: Enhancing mathematics learning with open-ended questions. *The Mathematics Teacher*, 88(6):496–499, 1995.
- [7] A. Iglesias, P. Martínez, R. Aler, and F. Fernández. Learning teaching strategies in an adaptive and intelligent educational system through reinforcement learning. *Applied Intelligence*, 31(1):89–106, 2009.
- [8] S. Levine, A. Kumar, G. Tucker, and J. Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.
- [9] F. S. Melo. Convergence of q-learning: A simple proof. *Institute Of Systems and Robotics, Tech. Rep*, pages 1–4, 2001.
- [10] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.
- [11] D. J. Nicol and D. Macfarlane-Dick. Formative

- assessment and self-regulated learning: A model and seven principles of good feedback practice. *Studies in higher education*, 31(2):199–218, 2006.
- [12] H. W. Park, I. Grover, S. Spaulding, L. Gomez, and C. Breazeal. A model-free affective reinforcement learning approach to personalization of an autonomous social robot companion for early literacy education. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 687–694, 2019.
- [13] S. Reddy, S. Levine, and A. Dragan. Accelerating human learning with deep reinforcement learning. In *NIPS'17 Workshop: Teaching Machines, Robots, and Humans*, pages 5–9, 2017.
- [14] J. P. Rowe and J. C. Lester. Improving student problem solving in narrative-centered learning environments: A modular reinforcement learning framework. In *International conference on artificial intelligence in education*, pages 419–428. Springer, 2015.
- [15] D. Shawky and A. Badawi. A reinforcement learning-based adaptive learning system. In *International Conference on Advanced Machine Learning Technologies and Applications*, pages 221–231. Springer, 2018.
- [16] V. J. Shute. Focus on formative feedback. *Review of educational research*, 78(1):153–189, 2008.
- [17] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Preliminary Experiments with Transformer based Approaches To Automatically Inferring Domain Models from Textbooks

Rabin Banjade
University of Memphis
rbnjade1@memphis.edu

Priti Oli
University of Memphis
poli@memphis.edu

Lasang Jimba Tamang
University of Memphis
ljtamang@memphis.edu

Vasile Rus
University of Memphis
vrus@memphis.edu

ABSTRACT

Domain modeling is a central component in education technologies as it represents the target domain students are supposed to train on and eventually master. Automatically generating domain models can lead to substantial cost and scalability benefits. Automatically extracting key concepts or knowledge components from, for instance, textbooks can enable the development of automatic or semi-automatic processes for creating domain models. We explore in this work the use of transformer based pre-trained models for the task of keyphrase extraction. Specifically, we investigate and evaluate four different variants of BERT, a pre-trained transformer based architecture, that vary in terms of training data, training objective, or training strategy to extract knowledge components from textbooks for the domain of intro-to-programming. We report results obtained using the following BERT-based models: **BERT**, **CodeBERT**, **SciBERT** and **RoBERTa**.

Keywords

Domain modeling, knowledge component extraction, key phrase extraction, transformer based models, Intelligent Tutoring System

1. INTRODUCTION

Computer-based adaptive instructional technologies, our focus, must have a representation of the target domain, i.e., a domain model. Hence, there is a need for domain modeling, which specifies the key knowledge components or units of knowledge that students have to master in a target domain, such as physics, biology, or computer programming. Furthermore, a domain model should include a structure that specifies the relationship among the knowledge components, typically in the form of a prerequisite knowledge structure suggesting which basic concepts must be mastered before

more complex concepts that rely on the basic concepts, e.g., addition should be mastered before multiplication [14, 17, 8]. The prerequisite knowledge structure of a domain model implies a specific trajectory or trajectories towards mastery that students must follow. According to some, a domain model should also include links to related learning objects, i.e., instructional tasks, which help students practice to master the targeted concepts. More recently, A broader view of the domain model has been argued, which should include all the key concepts, skills, ideas, principles, and the values, identity, and epistemology of the community of experts or professionals active in the target domain [1].

Domain models can be developed from different information sources: experts, textbooks (written by domain and pedagogical experts), and learner performance data. Expert-driven approaches to domain modeling are expensive, time-consuming, and not very scalable within and across domains. To overcome these challenges, automated or semi-automated approaches are much needed. This work explores such novel automated methods for domain model discovery from textbooks, specifically for the target domain of intro to computer programming. In particular, we explore to what extent the process can be automated. While the proposed methods are fully automated, their output is not perfect, which means a human expert must be involved to curate the output before being used in an existing adaptive instructional system. Nevertheless, this semi-automated process is much more cost-effective and scalable than the manual approach to building domain models. It should be noted that another source of information for domain modeling for programming that has been explored in the past is code itself, e.g., using a Java parser [18, 38, 15].

There are two advantages of using textbooks to extract domain models for programming compared to using code only. First, textbooks contain both code examples and textual explanations of the concepts, which is advantageous for higher-level concepts such as sorting, which is challenging to infer from a block of code that implements it. Second, concepts extracted directly from code are difficult to interpret and are often programming language-specific constructs. A textual description can accompany the grammar of a programming language in the form of comments or descriptions, but this requires substantial additional expert involvement. Build-

R. Banjade, P. Oli, L. J. Tamang, and V. Rus. Preliminary experiments with transformer based approaches to automatically inferring domain models from textbooks. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 667–672, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853051>

ing more interpretable automated ways to extract domain models from source code is an interesting research topic beyond this paper’s scope. In this work, we experiment with a pretrained transformer-based model to extract key concepts or knowledge components from computer science (intro to programming) textbooks. We use the code examples in textbooks, but we do not parse them to extract candidate key concepts but to rank the concepts extracted from the textual explanations.

Transformer-based pretrained models, which are trained on massive unlabelled text collections, have been successful in various NLP tasks such as keyphrase extraction [3, 33] which is relevant to this work. However, with the rise of many pretrained models for various specific tasks, there is a need to explore which of these pretrained models are helpful for what tasks. In this work on domain model discovery for computer programming, we explore an overgeneration-and-ranking approach for keyphrase extraction using four pretrained transformer models: BERT[11], CodeBERT[13], RoBERTa [25], and SciBERT [2]. These pretrained models vary in different aspects, such as training data and training mechanisms. For instance, BERT is trained on general domain corpora such as news articles and Wikipedia, RoBERTa is trained on a larger dataset of English language corpora consisting of books, news, web text; SciBERT is trained on papers on computer science and biomedical domain whereas CodeBERT is pretrained in NL-PL (Natural language-Programming language) pairs for multiple programming languages. In this study, we experiment with embeddings obtained from each of these methods to evaluate and compare domain-specific models such as CodeBERT and SciBERT versus models such as Roberta and BERT trained on general corpora for the tasks of keyphrase extraction and subsequently domain modeling.

The paper is organized as follows. Section 2 discusses relevant works to domain modeling. In section 3, we detail the methodology followed to generate domain models from textbooks. Section 4 presents the evaluation dataset and the metrics used to evaluate the performance of the pretrained models. A conclusion section follows the results.

2. RELATED WORK

Our work focuses on extracting key knowledge components from textbooks using embeddings obtained from transformer based pretrained models in an unsupervised manner. This section presents most relevant prior works to concept extraction, unsupervised keyphrase extraction, and pretrained models in NLP.

Concept extraction or identifying important concepts that a learner should master has been studied for various applications in the educational domain, such as concept-based textbook indexing (adaptive hyperbook for constructive teaching, Elm-art [5]), concept prediction [19], and concept hierarchy creation [36]. Concept extraction is related to keyphrase extraction, which is extracting the most important concepts in a given document. Keyphrase extraction has been explored using different approaches: rule-based, supervised, and unsupervised including deep-learning [30]. Typically, keyphrase extraction consists of two steps: candidate generation and ranking. The first step extracts key concepts

based on heuristics, such as all noun phrases, while the second step ranks the extracted candidate phrases based on scoring rubric that indicates the importance of the candidate phrase for the document and/or goal. In our work, we use unsupervised embedding-based ranking methods for candidate concepts generated from sections or subtopics in chapters in intro-to-programming textbooks. We consider a subtopic as a reference document when assessing the importance of each candidate key concept.

Existing unsupervised keyphrase extraction methods can be broadly categorized as statistics based such as TF-IDF, e.g., KP-Miner [12], graph-based such as TextRank [27], SingleRank [35] and topic-based methods such as TopicRank [4] even though many of the works use a combination rather than a single approach. Recent advances in representational methods of words, phrases and documents like Word2Vec [28], Doc2vec [20], and Sent2vec [29] led to novel ranking methods for keyphrase extraction [3, 23, 33]. For instance, EmbedRank [3] uses sentence embeddings based on Doc2vec or Sent2vec to represent candidate phrases and the document in the same high dimensional vector space based on which a ranking of the candidate key concepts is obtained using the cosine similarity score between the embedding vectors of the candidate phrases and the documents. Our work is similar in the sense that we use pretrained embeddings for keyphrase extraction in the context of domain modeling.

Embeddings obtained using transformer [34] based pretrained models [31, 32, 37] have shown dramatic improvements in various tasks in different domains such as software engineering, computer vision, and education. The reason behind this improvement is the high-quality semantic representations. Models such as BERT have been trained on different domain-specific data, some examples being BioBERT [21] in biology, legalBERT [7] in legal documents, SciBERT in scientific articles, and CodeBERT on code and natural language text pairs. To the best of our knowledge, experiments on how these embeddings affect the downstream task of knowledge component extraction and consequently on domain modeling have not been done before. In this work, we experimented with embeddings obtained from four BERT models, BERT, CodeBERT, SciBERT, and RoBERTa, for knowledge component extraction and the larger task of domain modelling.

3. METHODOLOGY

Our method to extract knowledge components from intro-to-programming textbooks is based on pretrained embeddings inferred from various BERT-based models. To evaluate the automated methods, we annotated a dataset by selecting key concepts for each section in five randomly selected chapters in two textbooks. As noted, we rely on an overgeneration-and-ranking approach for key concept extraction. The embeddings are mainly used to rank the key candidate phrases. Since textbooks for intro-to-programming contain both code examples and related explanatory text, we can use bimodal pretrained models such as CodeBERT trained on both text and code. Even though we provide code to the CodeBERT model as input, the model is used only for ranking candidate keyphrases, i.e., our approach and evaluation is based on candidate concepts generated from the textual parts of the textbooks. We could consider statements, code blocks,

and code comments as sources of candidate concepts. However, we limit the scope of this work to candidate concepts from the textual part of intro to computer programming textbooks.

In this section, we explain the preprocessing steps, such as candidate concept extraction, phrase and document embedding generation, and candidate concept ranking. We also discuss the performance metrics of specificity and relevancy. Before applying methods related to candidate concept extraction, we resolve pronouns in the text to boost our knowledge component extraction methodology by resolving pronouns such as 'it', 'this', and 'their' using a pretrained deep learning model [22] based on span ranking architecture.

3.1 Candidate Concept extraction

The step of candidate concept extraction consists of noun phrase extraction and filtering. We used Stanford CoreNLP Tools ¹ for tokenizing, part-of-speech tagging, and noun phrase chunking. We only considered noun phrases which are unigrams (one token), bigrams (two consecutive tokens), trigrams (three consecutive tokens), and quadgrams (four consecutive tokens), for candidate generation.

3.2 Phrase and Document Embedding

From the prior step of generating 3.1 candidate phrases we obtain a tokenized form of a document D represented as $D = t_1, t_2, t_3, \dots, t_N$ where t_n represent tokens. We also obtain a list of candidate phrases C_0, C_1, \dots, C_N . Based on this, we then obtain contextualized embeddings for each of the tokens as shown by the Equation 1.

$$E_1, E_2, E_3, \dots, E_N = Model(t_1, t_2, t_3, \dots, t_N) \quad (1)$$

where E_n represents embeddings of each token, and the model refers to any of the four pretrained models we chose: BERT, CodeBERT, SciBERT or RoBERTa.

Each document embedding is obtained using average pooling across all the tokens of the document. Similarly, to obtain embedding for each candidate phrase from a document, we average across embeddings of tokens. Although the best pooling strategy is still an area of active research, we opted for the average pooling strategy as it has shown better performance than using the output of the first token [CLS] for different tasks[9].

3.3 Ranking

We have experimented with a number of ranking and performance metrics as presented in this section.

3.3.1 Cosine Similarity

Once the embeddings for each document and phrase are obtained, we compute the cosine similarity (normalized dot product) between the phrase and document embedding vectors. The cosine similarity scores for candidate phrases capture the semantic 'relatedness' or 'closeness' of a phrase to the underlying document. Those scores are used to rank candidate phrases.

¹<https://nlp.stanford.edu/software/tagger.shtml>

3.3.2 Adjusted Maximal Marginal Relevance To Balance General and Specific Concept Ranking

Our goal is to extract all important knowledge components of a target domain. A challenge to this coverage problem is that key concepts can have different granularity levels, making it more challenging to design a single metric that ranks more general and more specific concepts highly.

For instance, in the context of a chapter focusing on the concept of 'loop', some of the important key concepts are 'for loop' or 'while loop', which are broad, high-level types of loops. Nonetheless, a more specific concept such as a nested loop or loop continuation condition is important too.

In order to extract both the general and specific concepts, candidate key phrases must be similar to the current document and less relevant to other documents. Furthermore, in order to rank higher topic-specific concepts so that top k candidates consist of more topic-specific concepts, we modified the Maximal Marginal Relevance (MMR) [6] metric, which was used initially in information retrieval and text summarization to control relevancy and diversity of retrieved documents. The embedding-based keyphrase extraction method proposed by Bennani-Smires et al. [3] used a modified version of MMR to tackle redundant key phrases from a document. In our case, we modify MMR to balance through the control parameter λ general and specific key candidate phrases.

As indicated in the equation 2, we balance the similarity of a candidate phrase C_i to the current document doc , which is captured by the first term on the right-hand side of the equation, with its similarity with other documents which is indicated by the second term that represents the similarity between the candidate phrase C_i from all the other N documents in the corpus.

If a candidate phrase is highly relevant to a document other than the current document, the similarity measure with the current document is penalized, indicating the term is not specific enough to the current document.

$$MMR_{adjusted} := \arg \max_{C_i \in K} \left[\lambda \cdot \text{sim}_1(C_i, doc) - (1 - \lambda) \max_{doc_j \in N \setminus doc} \text{sim}_2(C_i, doc_j) \right] \quad (2)$$

4. EXPERIMENT AND RESULTS

As already noted, we experimented with four pretrained models and evaluated their performance on a dataset that we built.

4.1 Dataset

The evaluation dataset was built based on sections from two intro-to-programming textbooks, "Introduction to JAVA programming" [24] and "JAVA How to program" [10]. From each of the two textbooks, we randomly selected 20 sections that focus on some specific concepts such as 'while-loop,' 'sorting arrays,' and 'exception handling overview.'

For each of the 20 sections, two computer science graduate students manually extracted the key concepts using a

two-phase annotation scheme. In the first phase, each annotator selected key concepts. The inter-annotator agreement in this phase was 0.7 as measured by Cohen’s Kappa [26]. Then, annotators discussed iteratively and extensively until a final agreement was reached. The resulting key concepts form our gold standard for evaluating the proposed methods. We have also asked each annotator to rank the gold standard concepts. This ranking is used for another performance metric that compares the automated ranking to the human ranking.

4.2 Evaluation Metrics

We evaluated the four pretrained models using two approaches. First, we evaluated the key concept extraction using precision, recall, and the F-score at rank k for $k = 5, 10, 15$. This evaluation approach is widely used in key-phrase extraction systems[30]. The other evaluation approach is based on the Normalized Discounted Cumulative Gain at p ($NDCG_p$) [16]. $NDCG_p$ compares the target ranking to the positions that key concepts occupy in the gold standard ranking and penalizes any mismatches. We opt for this evaluation to evaluate the extracted concepts based on $MMR_{adjusted}$ ranking method described earlier.

4.3 Results

First, we report results based on precision, recall, and F-score for top k ranks where $k = 5, 10, 15$ using cosine similarity-based ranking. We also report results for $NDCG_p$ where $p = 10$ using adjusted MMR metric with $\lambda = 0.5$ and $\lambda = 1$. The $NDCG_p$ ranking results obtained using $\lambda = 1$ for adjusted MMR metric is same as ranking using cosine similarity only. Since the average number of key concepts per section in the gold standard is 11, we chose $p = 10$ for normalized discounted cumulative gain ($NDCG$) reporting.

Table 1: Precision, recall and Fscore at 5,10,15 for knowledge component extraction

K	Model	P	R	F
5	BERT	0.66	0.423	0.499
	CodeBERT	0.54	0.342	0.405
	RoBERTa	0.5	0.288	0.354
	SciBERT	0.58	0.41	0.480
10	BERT	0.578	0.659	0.597
	CodeBERT	0.456	0.521	0.473
	RoBERTa	0.522	0.589	0.538
	SciBERT	0.533	0.608	0.55
15	BERT	0.516	0.801	0.615
	CodeBERT	0.367	0.578	0.442
	RoBERTa	0.434	0.673	0.516
	SciBERT	0.483	0.745	0.576

Table 1 shows Precision, recall and F-scores for different pretrained embeddings for $k = 5, 10, 15$. We can see that BERT has the highest precision, recall, and F-score for all the values of k . Similarly, SciBERT yields better performance compared to RoBERTa and CodeBERT. We can observe that even though CodeBERT is trained in NL-PL pairs for different programming languages, it does not provide any advantage over BERT. This might be because the training data for CodeBERT, which is specific to the code in Github, relates to higher-level software engineering concepts than intro-to-programming concepts. Also, the natural language

Table 2: Normalized Discounted Cumulative Gain($NDCG_{10}$) using $MMR_{adjusted}$ for $\lambda = 1$ and $\lambda = 0.5$.

Model	NDCG@10	
	$\lambda = 1$	$\lambda = 0.5$
BERT	0.83	0.87
CodeBERT	0.75	0.73
SciBERT	0.81	0.80
RoBERTa	0.76	0.77

texts used for training CodeBERT is mainly code documentation which might not be directly relevant to basic programming concepts. RoBERTa, even though trained with a different strategy and more data, does not show any better performance than BERT and SciBERT.

Also, it is evident from the results that SciBERT, trained in scholarly documents from computer science and biomedical domain, performs on par with BERT. The results show that even though BERT is not trained on any domain-specific data, it performs better than other models trained with more domain-specific or more data.

Table 2 shows the $NDCG$ score using $MMR_{adjusted}$ in equation 2. The results are obtained using $\lambda = 1$ which is equivalent to using cosine similarity only and $\lambda = 0.5$ which gives equal importance to the general and specific components in the adjusted MMR metric. As shown in the results, BERT outperforms other embedding methods similar to the evaluation results shown in Table 1 based on precision, recall, and F-measure. The value of $NDCG$ of all the models except for CodeBERT was higher when ranking was done using $\lambda = 0.5$ for $MMR_{adjusted}$ compared to value of λ set to 1.

5. DISCUSSION AND CONCLUSION

Our results indicate that we can use pretrained models for domain model extraction. We also noticed that even though models like CodeBERT and SciBERT are trained on domain-specific data, they did not provide any advantage mainly due to the nature of their training data. Our experiments also evaluated ranking based on Maximal Marginal Relevance to balance general and specific concepts. The main idea behind the evaluation was to check if adjusted MMR as we propose, ranks topic-specific concepts higher. Even though control parameter λ can be trained across different documents to get a more precise value based on cumulative gain we used 0.5 for key concept extraction using a relevance score based on Mean Marginal Relevance to provide equal importance to general and specific concepts. Evaluation based on Table 2 shows that $MMR_{adjusted}$ ranking obtained when $\lambda = 0.5$ is closer to gold-standard ranking or preferred ranking by human annotators.

Acknowledgement

This work has been supported by two NSF awards: Learner Data Institute (NSF award 1934745) and CSEdPad: Investigating and Scaffolding Students’ Mental Models during Computer Programming Tasks to Improve Learning, Engagement, and Retention (NSF award 1822816). The opinions, findings, and results are solely the authors’ and do not reflect those of NSF.

6. REFERENCES

- [1] R. Banjade, P. Oli, L. J. Tamang, J. Chapagain, and V. Rus. Domain model discovery from textbooks for computer programming intelligent tutors. *The International FLAIRS Conference Proceedings*, 34, Apr. 2021.
- [2] I. Beltagy, K. Lo, and A. Cohan. Scibert: A pretrained language model for scientific text. *arXiv preprint arXiv:1903.10676*, 2019.
- [3] K. Bennani-Smires, C. Musat, A. Hossmann, M. Baeriswyl, and M. Jaggi. Simple unsupervised keyphrase extraction using sentence embeddings. *arXiv preprint arXiv:1801.04470*, 2018.
- [4] A. Bougouin, F. Boudin, and B. Daille. Topicrank: Graph-based topic ranking for keyphrase extraction. In *International joint conference on natural language processing (IJCNLP)*, pages 543–551, 2013.
- [5] P. Brusilovsky, E. Schwarz, and G. Weber. Elm-art: An intelligent tutoring system on world wide web. In *International conference on intelligent tutoring systems*, pages 261–269. Springer, 1996.
- [6] J. Carbonell and J. Goldstein. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 335–336, 1998.
- [7] I. Chalkidis, M. Fergadiotis, P. Malakasiotis, N. Aletras, and I. Androutsopoulos. Legal-bert: The muppets straight out of law school. *arXiv preprint arXiv:2010.02559*, 2020.
- [8] H. Chau, I. Labutov, K. Thaker, D. He, and P. Brusilovsky. Automatic concept extraction for domain and student modeling in adaptive textbooks. *International Journal of Artificial Intelligence in Education*, 31(4):820–846, 2021.
- [9] H. Choi, J. Kim, S. Joe, and Y. Gwon. Evaluation of bert and albert sentence embedding performance on downstream nlp tasks. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 5482–5487. IEEE, 2021.
- [10] P. J. Deitel and H. M. Deitel. *Java*. Pearson, 2015.
- [11] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [12] S. R. El-Beltagy and A. Rafea. Kp-miner: Participation in semeval-2. In *Proceedings of the 5th international workshop on semantic evaluation*, pages 190–193, 2010.
- [13] Z. Feng, D. Guo, D. Tang, N. Duan, X. Feng, M. Gong, L. Shou, B. Qin, T. Liu, D. Jiang, et al. Codebert: A pre-trained model for programming and natural languages. *arXiv preprint arXiv:2002.08155*, 2020.
- [14] I. Goldin, P. I. Pavlik Jr, and S. Ritter. Discovering domain models in learning curve data. *Design Recommendations for Intelligent Tutoring Systems: Volume 4-Domain Modeling*, 4:115–126, 2016.
- [15] R. Hosseini and P. Brusilovsky. Javaparser: A fine-grain concept indexing tool for java problems. In *CEUR Workshop Proceedings*, volume 1009, pages 60–63. University of Pittsburgh, 2013.
- [16] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446, 2002.
- [17] K. R. Koedinger, A. T. Corbett, and C. Perfetti. The knowledge-learning-instruction framework: Bridging the science-practice chasm to enhance robust student learning. *Cognitive science*, 36(5):757–798, 2012.
- [18] A. N. Kumar. Using enhanced concept map for student modeling in programming tutors. In *FLAIRS Conference*, pages 527–532, 2006.
- [19] I. Labutov, Y. Huang, P. Brusilovsky, and D. He. Semi-supervised techniques for mining learning outcomes and prerequisites. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 907–915, 2017.
- [20] J. H. Lau and T. Baldwin. An empirical evaluation of doc2vec with practical insights into document embedding generation. *arXiv preprint arXiv:1607.05368*, 2016.
- [21] J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So, and J. Kang. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240, 2020.
- [22] K. Lee, L. He, M. Lewis, and L. Zettlemoyer. End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197, 2017.
- [23] X. Liang, S. Wu, M. Li, and Z. Li. Unsupervised keyphrase extraction by jointly modeling local and global context. *arXiv preprint arXiv:2109.07293*, 2021.
- [24] Y. D. Liang. *Introduction to Java programming and data structures*. Pearson Education, 2018.
- [25] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.
- [26] M. L. McHugh. Interrater reliability: the kappa statistic. *Biochemia medica*, 22(3):276–282, 2012.
- [27] R. Mihalcea and P. Tarau. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411, 2004.
- [28] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space, 2013.
- [29] M. Pagliardini, P. Gupta, and M. Jaggi. Unsupervised learning of sentence embeddings using compositional n-gram features. *arXiv preprint arXiv:1703.02507*, 2017.
- [30] E. Papagiannopoulou and G. Tsoumakas. A review of keyphrase extraction. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 10(2):e1339, 2020.
- [31] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics*:

- Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.
- [32] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever. Improving language understanding by generative pre-training. 2018.
- [33] Y. Sun, H. Qiu, Y. Zheng, Z. Wang, and C. Zhang. Sifrank: a new baseline for unsupervised keyphrase extraction based on pre-trained language model. *IEEE Access*, 8:10896–10906, 2020.
- [34] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
- [35] X. Wan and J. Xiao. Single document keyphrase extraction using neighborhood knowledge. In *AAAI*, volume 8, pages 855–860, 2008.
- [36] S. Wang, C. Liang, Z. Wu, K. Williams, B. Pursel, B. Brautigam, S. Saul, H. Williams, K. Bowen, and C. L. Giles. Concept hierarchy extraction from textbooks. In *Proceedings of the 2015 ACM Symposium on Document Engineering*, pages 147–156, 2015.
- [37] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32, 2019.
- [38] M. Yudelson, R. Hosseini, A. Vihavainen, and P. Brusilovsky. Investigating automated student modeling in a java mooc. *Educational Data Mining 2014*, pages 261–264, 2014.

E-learning Preparedness: A Key Consideration to Promote Fair Learning Analytics Development in Higher Education

Jinnie Shin
University of Florida
jinnie.shin@coe.ufl.edu

Okan Bulut
University of Alberta
bulut@ualberta.ca

Wallace N. Pinto Jr.
University of Florida
wallace.pintojun@ufl.edu

ABSTRACT

E-learning preparedness in higher education is an important aspect that determines students' interaction behaviors within online learning environments. This study is primarily motivated by the growing evidence pointing to the importance of constructing fair and unbiased learning analytics in higher education. The primary goal of this study was to examine the impact of potentially bias-driven variables in predictive learning analytics models. We used an empirical data set collected from 123 undergraduate students who participated in a remote asynchronous course in Fall 2021. Our study adopted various statistical and machine learning techniques to remedy the biased prediction behaviors in learning analytics. First, we conducted a path analysis to evaluate the connection between students' e-learning preparedness and their interaction within the e-learning system. We simulated a large synthetic dataset from the empirical dataset and adopted two fair artificial intelligence algorithms—Seldonian and Adversarial Networks algorithms. Our findings indicated that e-learning preparedness is significantly related to the commonly adopted temporal features in learning analytics, such as time-use entropy ($b=.449, p<.001$) and assignment submission time ($b=-.587, p<.001$), and indirectly associated with students' course performance ($b=.219, p<.001$). Both algorithms indicated that bias-reduced algorithms performed comparably worse than the naive algorithms in predicting students' performance (f1-score=0.72); however, they produced less biased predictions after taking students' e-learning preparedness into account. A significant shift in model weights were observed, where the models tended to emphasize the importance on the variety of activity sequence than the temporal information.

Keywords

Fairness in AI, Learning Analytics, E-learning preparedness, Seldonian Algorithm, Adversarial Networks

1. INTRODUCTION

The sudden instructional and pedagogical changes that the COVID-19 pandemic has brought in higher education have been tremendous. The recent U.S. statistics indicated that more than 52% of college students had to participate in remote learning during the year 2019-2020. Many universities and college-level courses that are designed for face-to-face in-person learning, transitioned online. Instructors prepared asynchronous or synchronous remote learning courses with lecture videos and learning materials to

accommodate the changes. Learning analytics systems were introduced to support managing distance learning for higher education institutions. Learning analytics frameworks adopt various statistical and machine learning techniques to enrich evidence-based instructions [1]. Many of the previously introduced learning analytics systems in higher education focused on detecting at-risk students [5, 16] and predicting student success [17]. Higher education institutions gained the capacity to gather, manage, and access student learning information more closely with the wide adoption of distance learning. The adoption of various learning analytics systems opened opportunities for an in-depth understanding of students' learning progress.

Despite the interesting benefits that learning analytics systems brought to higher education institutions, a growing concern evolved in the fairness and equitable uses of learning analytics [8, 10, 23]. One of the many underlying fairness and bias-related issues in learning analytics concerns the incomplete consideration and critical evaluation of the nature of the data and data collection [25]. For instance, with the sudden shift to remote learning, a considerable number of students faced significant challenges that arise from a lack of preparedness for distance learning [2, 6, 19, 21, 22]. Faculties and instructors indicated some of the challenges they faced during this period, such as students' lack of basic resources and equipment to participate in e-learning, and disruptions to learning time due to poor internet connections [29].

Likewise, students may require a more reliable setup for learning at home with the appropriate gears and equipment to participate in remote learning. The varying level of physical and psychological preparedness for e-learning showcases the important underlying nature of data and data collection in learning analytics, especially during the rapid transition to online learning in higher education institutions. For instance, reaction-time or temporal features are commonly adopted for student success prediction [14, 15]. They evaluate time-related students' interaction attributes, such as how rapidly students react to the uploaded learning resources. The reaction-time features are often positively associated with students' self-regulated learning behaviors and improved learning outcomes in many learning analytics algorithms [20, 27]. This study takes a different view on how "fairness" could be acquired in learning analytics for higher education. Fairness in learning analytics was often discussed in the previous studies with the light on students' unique personal and demographic backgrounds (e.g., gender). Instead, we focus on how the learning analytics could potentially make biased decisions or predictions for the students with the lack of preparedness in e-learning. The following research questions were addressed in this study: (1) *Does students' level of e-learning preparedness relate to their interaction behaviors in an online course platform?* (2) *Does students' level of e-learning preparedness relate to their course performance in an online course?* and (3) *Does the bias-reduced prediction model, which*

J. Shin, O. Bulut, and W. N. P. Jr. E-learning preparedness: A key consideration to promote fair learning analytics development in higher education. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 673–678, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853111>

takes students' e-learning preparedness into account, achieve significantly lower accuracy?

2 RELATED WORK

2.2 E-learning Preparedness

An exhaustive review of the past 20 years of the literature revealed interesting underlying dimensions of students' e-learning preparedness in higher education, although there is no consensus on what exactly preparedness or readiness is. Earlier studies focused on students' "readiness" in online learning. For instance, Smith et al. [24] identified students' psychological factors and personal attributes, such as their comfort with e-learning and self-management of learning, as two primary underlying factors which define students' readiness in e-learning environments. Their findings were replicated and confirmed by other researchers (e.g., [3]). Watkins et al. [28] expanded the definition of e-learning readiness with six dimensions, including the two unique physical environmental attributes, such as the "Access to Technology" and "Online Audio/Video", as well as psychological aspects as "Online Skills and Relationships", "Motivation", "Internet Discussions", and "Importance to their Success". Similarly, Holsapple and Lee-Post [12] suggested that e-learning preparedness should be measured in the primary three dimensions of students' technical competencies, lifestyle aptitudes, and learning preferences. Parkes et al. [21] proposed a shift in focus between readiness and preparedness. They focused on the "necessary" skills and competencies students need to be prepared for e-learning. For instance, students' competencies in "managing learning in the online environment", "interacting with the e-learning content", and the "e-learning communities", were identified as important dimensions to define one's preparedness for e-learning.

2.3 Fair Prediction in AI

The following two frameworks are introduced and adopted in our study to demonstrate the potential bias that presents in algorithm.

2.3.2 Seldonian Algorithm

The Seldonian algorithm [26] introduces a three-stage framework to prevent machine learning algorithms from making biased outcomes. The algorithm takes the data (D) and partitions it into two sets of D_1 and D_2 . Using the first part of the dataset, the algorithm selects a solution candidate, θ_c , which is likely to pass the algorithm bias safety test. Then, the second part of the data, D_2 is used as a safety set. The Seldonian algorithm allows safety constraints to be represented as $(g_i, \delta_i) \forall_i \in m$. $g: \theta \rightarrow R$ is a function that quantifies the safety of the solution and is the permissive probability of returning an unsafe solution. The safety constraint ensures that the algorithm could achieve $g_i(\theta, D) \leq 0$ to ensure a safe choice of solution candidate. When only one constraint is given, this constraint can be understood as an equalized odds constraint [11]. The equalized odds constraint is defined based on the true positive rate (or TPR) across groups that are designed to evaluate and compare the degree of bias in model performance. Assume that A is the sensitivity feature to create two groups (0 = unprepared, 1 = prepared in e-learning) and Y is a predicted outcome variable (1 = success in the course). The equalized odds are satisfied when both the true-positive and the false-positive rates are equal across the two groups (equation 2).

$$\begin{aligned} Pr(\hat{Y} = 1|A = 0, Y = 1) &= Pr(\hat{Y} = 1|A = 1, Y = 1) \\ Pr(\hat{Y} = 0|A = 0, Y = 0) &= Pr(\hat{Y} = 0|A = 1, Y = 0) \end{aligned} \quad (2)$$

Using the equalized odds constraint, the constraint, $g(\theta)$, can be expressed when we decide to create an equalized odds score that is smaller than 0.05. In other words, this constraint has the upper bound of 0.05 to account for the absolute differences between the accuracy in predictions between the two groups.

2.3.3 Adversarial Neural Networks

Adversarial networks are supervised learning techniques that systematically associates the uncertainties in the data generation process to represent the real-world setting more accurately. Given a set of probabilities, $Pr(X, Y, Z)$, which consists of the data X target, Y, and the contextual sensitive parameter, Z, in fair AI prediction, we aimed to develop a solution that maps $f(X) = Y$, with the classifier f that is robust to the value of Z. This process can be learned efficiently using the adversarial networks [9]. For fair AI models, we could constrain the predictive classifier f to satisfy a certain constraint which has to do with the sensitive parameter Z. The goal of our classifier is to learn the parameters θ_f which map $f: \chi \rightarrow S$. Hence, we minimize the cross-entropy loss function, $L_y(\theta_f)$ to obtain the parameters, θ_f . In the case of student performance prediction, $s \in S$ represents students' performance outcome. The initial inference of the classifier will be defined based on $f(X; \theta_f)$ and the value of $z \in Z$ will be unknown to the model now. This indicates that the model outputs the equal conditional probability for any $z, z' \in Z$ and $s \in S$ (equation 3).

$$Pr(f(X; \theta_f) = s|z) = Pr(f(X; \theta_f) = s|z') \quad (3)$$

3 METHODS

3.2 Data

Our dataset was collected from a total of 123 undergraduate students who are enrolled in an educational assessment course in a western Canadian university in Fall 2020. This course was traditionally delivered as an in-person course but transitioned to an asynchronous online course in Fall 2020. At the beginning of the semester, the students were provided with a "welcome survey" questionnaire. The questionnaire included four items evaluating students' physical and psychological readiness and preparedness for an online course (see Appendix). We recoded the responses to define a variable representing "e-learning preparedness" in an online learning environment. We hypothesized that the students who are physically well-prepared for the online course would currently be located at the same time-zone, have devices that are equipped with the required functionality to participate and interact with the course materials, and have reliable devices that are portable. This was based on the course outline and the requirements related to the course materials and assignments. Once the physical/environmental preparedness score is computed, it is combined with the psychological preparedness score. Final preparedness is represented as a binary score of those who are above (1) or below (0) average in environmental and psychological preparedness for online learning (see Tables 1 and 2).

3.3 Interaction Log Data

To understand students' learning behaviors within the online course module, we extracted a comprehensive list of students' behaviors and selected the deterministic features to construct a prediction model. To replicate commonly extracted features in prediction-focused learning analytics systems, we used a commonly adopted open-source automated feature engineering tool called *Featuretools*. It performs Deep Feature Synthesis to extract aggregative and transformative features from the interaction log data. The extracted features represent descriptive instances (e.g.,

absolute, count, mean, mode), as well as the advanced interactions (e.g., entropy, time since previous). This feature analysis framework was selected as they are frequently adopted in learning analytics [4]. In addition, we extracted a list of n-gram-based log activity features. The features represent a sequence of instructional activities that students frequently perform. A total of 132 unique activities are captured and classified into 14 overall categories. The n-gram-based features represented a frequent sequence of actions that students partake among these 14 activity categories. The feature and the values were extracted using a TF-IDF vectorization.

3.4 Path Analysis Model

We aimed to understand the association of students' e-learning preparedness with their interaction activities which are represented by the log features. We constructed a path analysis model using the empirical data collected from the undergraduate participants (n=123). The hypothesized model path model first controlled for the effect of students' familiarity with the course content (Familiarity) on the students' course outcome (Summative). This allowed us to evaluate the relationship between e-learning preparedness with other variables while controlling for the effect of familiarity. The path model evaluated the associations between the preparedness and interaction log activities. Two interaction log variables were selected from the final feature sets extracted in the previous stage. Last, the two interaction log features were related to the students' formative and summative course performance. Our hypothesized model evaluated the importance of preparedness in understanding students' interaction in an online course as well as the relationship with the course outcomes.

3.5 Simulation Setting

We used the distributional characteristics of the variables and the correlations among the variables in our empirical dataset to simulate a larger-size dataset (N = 10,000). The data simulation process consisted of several steps. First, we obtained the distribution parameters for the variables included in the path analysis model. Second, we calculated the target correlation matrix based on the observed correlations in the empirical dataset. Third, we checked the lower and upper bounds of the pairwise correlations for the given distribution parameters to ensure that the target correlation matrix is within the bounds. Next, we simulated the dataset using the *SimMultiCorrData* package [7] in R. Last, we reviewed the generated variables and the maximum error between the final and target correlation matrices.

4 RESULTS

4.2 Features

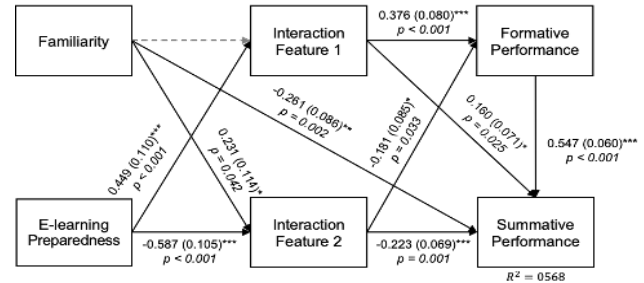
The survey results with the empirical dataset indicated that close to 32.5% of the students scored below the average in the total preparedness score. The rest of the students (67.5%) scored above the average. The e-learning preparedness score had a mean of 4.46 and a standard deviation of 1.61. The familiarity with the content with the total of 9 key concepts indicated the mean of 16.54 and the standard deviation of 6.55. The interaction log feature engineering using the Featuretools extracted a total of 137 variables. The n-gram tf-idf features extracted a total of 102 features to represent the frequency and the sequence of students' log actions. The feature dimension reduction process resulted in a total of 13 features.

4.3 Path Analysis Results

Figure 1 provides a diagram of the final path analysis model which is hypothesized and evaluated in our study. The path analysis results showed a good model-fit (CFI > 0.90, TLI > 0.90, RMSEA

<0.06, SRMR <0.09; [13]). The two variables representing students' interaction with the online course platform (Interaction Features 1 and 2) were also significantly associated with students' preparedness ($b=0.433$, $s.e.=0.073$, $p<0.001$; $b=-0.421$, $s.e.=0.074$, $p<0.001$). This was evaluated while controlling for the effect of students' familiarity with the course content. Interestingly, the familiarity of course content was significantly related to only one of the interaction log activities (interaction feature 1: $b=-0.022$, $s.e.=0.115$, $p=0.851$; interaction feature2: $b=0.231$, $s.e.=0.114$, $p=0.042$). This shows that the students' level of preparedness for e-learning is significantly related to their activities in online learning platforms when controlling for their familiarity with the material.

The level of interaction with the learning platforms were also significantly related to students' formative ($b=0.376$, $s.e.=0.080$, $p<0.001$; $b=-0.181$, $s.e.=0.085$, $p=.033$) and summative course performance ($b=0.160$, $s.e.=0.071$, $p=.025$; $b=-0.223$, $s.e.=0.069$, $p=.001$), respectively. The findings also showed that the interaction log activities which were highly associated with students' "E-learning preparedness" were also highly related to students' formative and summative performance. In summary, we observed a significant total indirect effect of e-learning preparedness on summative performance score ($b=0.219$, $s.e.=0.048$, $p<0.001$) with the direct effect that is statistically not significant ($b=0.107$, $s.e.=0.060$, $p=0.073$). This shows that students' preparedness in e-learning may not be directly associated with their online learning performance when controlling for the students' familiarity with the course content. However, e-learning preparedness was highly associated with students' interaction activities in online learning, which were highly associated with the performance outcomes. In summary, the path analysis model demonstrated the importance of acknowledging the presence of varying levels of "e-learning preparedness"



Model-fit: CFI (0.987), TLI (0.938), RMSEA (0.081), SRMR (0.042).

Figure 1. Path Analysis Model Results with Significant Paths.

4.4 Fair AI Prediction Results

4.4.2 Logistic Regression Performance

Table 3 (see Appendix) provides a comparison of Seldonian model performance results. The classification performance results were provided based on the binary target of students' summative performance that is created using the quantile cut-offs (0.10 - 0.90). The findings showed that the bias-reduced logistic regression (LR) models based on students' "E-learning preparedness" may not perform as well as the unconstrained LR model. It showed that our logistic regression algorithm could achieve the highest accuracy and f1-scores in classifying the target variable across all quantile cut-offs. The best classification performance of the unconstrained LR model achieved the f1-score of 0.95. Seldonian models with the bigger equalized odds values performed better (acc=0.72, f1-score=0.76 vs. LR acc=0.76, f1-score=0.80).

4.4.3 Adversarial Networks Performance Results

Figure 2 provides diagrams that represent the distribution of the prediction of unprepared (yellow) and prepared (red) students when using the adversarial networks. The evaluation results were based on classifying students' summative performance with the 0.40 quantile as a cut-off. The initial performance of the biased classifier achieved an accuracy of 0.74, f1-score of 0.78, and the AUC-ROC score of 0.81. The classification distribution indicated that students from the unprepared category showed a noticeably higher probability to be less successful in the summative assessment (see Figure 2 - Unconstrained). By contrast, the well-prepared students showed a higher probability to be successful. Likewise, the classifier showed relatively biased classification results with the AUC-ROC score of 0.66 to classify students based on their preparedness. After a total of 100 epochs, we noticed that the adversarial algorithm with a constraint (or bias-reduced model) achieved an accuracy of 0.63, f1-score of 0.72, and AUC-ROC of 0.63 in classifying students based on their summative performance outcome. The final algorithm could no longer differentiate the students based on their e-learning preparedness (AUC-ROC 0.51). This indicates the final model was well-trained to remove the potential influence from the bias classification. The last figure in Figure 2 provides a visual representation of the final model classification distribution comparisons. It showcases how the two distributions converged. We noticed that the final bias-removed model could acquire a relatively good classification performance.

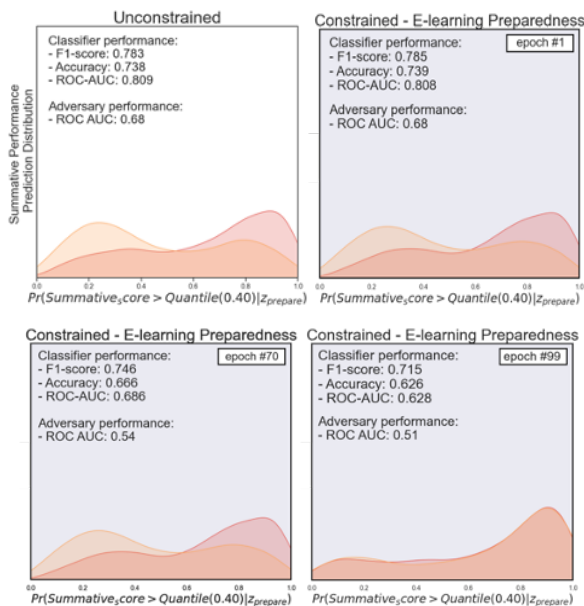


Figure 2. Unconstrained & Adversarial networks model results

We compared the final model weights of the biased model and the bias-reduced model. The relative impact of a total of 13 interaction log features (see Figure 3; Appendix) on classification was investigated using Shap [18]. The feature values (high or low, red, or blue) indicated the impact of the feature in producing higher or lower values in classification. Each dot (red or blue) represents the location of feature values in each sample. The x-axis represents the correlation between the final classification outcome and the feature values. The interaction log features were ranked in descending order based on their importance in classification. For instance, a high value in Feature 6 (red dots) was associated with the final classification (1=success) with a negative correlation (Shap values <0) in the biased classification model. Instead, the lower values

(blue dots) in Feature 6 were positively associated with the outcome classification in the biased model. By contrast, the lower values in Feature 10 were negatively correlated with the student success in the unconstrained model. Features 6 followed by the Features 10, 12, 4, 3, and 9 were identified as important and contributing features in the biased model to classify students based on their performance. Interestingly, the importance of features and direction of relationships with the outcome shifted quite drastically in the bias-reduced model. In the biased-constrained model, feature 2 was considered the most important, with its higher values (red) associated negatively with student success. Features 2 followed by the Features 10, 5, 4, and 6 were identified as important and contributing features in the bias-reduced model to classify students based on their performance. The direction of the relationship and the relative importance of Feature 10 remained the same for both models. Feature 10 represents the frequency of engaging in an activity related to learning resources (e.g., downloading).

5 DISCUSSION AND CONCLUSION

The primary goal of this study was to investigate the presence of the potentially biased behaviors of learning analytics models in higher education. We introduced empirical (N=123) and simulated (N=10,000) datasets of students' course logs to understand the biased-prediction behaviors of learning analytics models based on students' e-learning preparedness. Our study was conducted in two stages. First, we evaluated the association between students' e-learning preparedness and their online course performance outcomes. We specifically investigated the indirect (via the interaction log features) and direct relationship between the preparedness and summative performance outcomes (i.e., course exams). Students' interaction behaviors in the course were represented by the commonly adopted learning analytics features, such as their activity sequence and temporal information. Second, we investigated how the fair artificial intelligence (AI) models—Seldonian Algorithm and Adversarial Networks—could perform when attempting to reduce the bias in predicting students' performance outcomes. The two models were designed and implemented so that they could acknowledge students' varying levels of preparedness in participating in remote learning.

Our analytic framework using the path analysis model and two fair AI algorithms -- the Seldonian algorithm and the adversarial networks -- demonstrated efficient and effective means to evaluate the "e-learning preparedness" as a potential source of bias in learning analytics in higher education. We also visually demonstrated how the bias-reduced model shifted its model weights to emphasize the features that are potentially more robust to the (physically and psychologically) unprepared groups of students in remote learning. For future work, we aimed to generalize the findings with more empirical datasets in higher education settings, such as the e-learning interaction log datasets from different audiences and domains. We would like to examine how such dynamics differ as remote learning matures and students' preparedness (especially physical preparedness) changes with the consistent support from higher education institutions. Namely, our ultimate objective is to provide and promote less biased and more informative learning analytics in higher education, which could provide timely interventions to at-risk students. The findings from our study delineate the importance of careful consideration of the potentially biased source of information when constructing learning analytics models in a higher education environment. Our findings suggest that students' e-learning preparedness should be carefully considered to provide effective and diagnostic evaluation using learning analytics systems.

6 REFERENCES

- [1] Agarwal, R. and Dhar, V. 2014. Editorial—Big Data, Data Science, and Analytics: The Opportunity and Challenge for IS Research. *Information Systems Research* 25(3):443-448. <https://doi.org/10.1287/isre.2014.0546>.
- [2] Ana, A. 2020. Students' Perceptions of the Twists and Turns of E-learning in the Midst of the Covid 19 Outbreak. *Revista Romaneasca pentru Educatie Multidimensionala*, 12(1Sup2), 15-26. <https://doi.org/10.18662/rrem/12.1sup1/242>
- [3] Blankenship, R., and Atkinson, J. K. 2010. Undergraduate student online learning readiness. *International Journal of Education Research*, 5(2), 44-54.
- [4] Bosch, N., 2021. AutoML Feature Engineering for Student Modeling Yields High Accuracy, but Limited Interpretability. *Journal of Educational Data Mining*, 13(2), pp.55-79.
- [5] de Oliveira, C. F., Sobral, S. R., Ferreira, M. J., & Moreira, F. (2021). How Does Learning Analytics Contribute to Prevent Students' Dropout in Higher Education: A Systematic Literature Review. *Big Data and Cognitive Computing*, 5(4), 64.
- [6] Elumalai, K. V. et al. 2020. Factors affecting the quality of e-learning during the COVID-19 pandemic from the perspective of higher education students. *Journal of Information Technology Education: Research*, 19, 731-753. <https://doi.org/10.28945/4628>
- [7] Fialkowski, A. C. 2018. SimMultiCorrData: Simulation of correlated data with multiple variable types. R package version 0.2.2.
- [8] Gardner, J., Brooks, C. and Baker, R., 2019, March. Evaluating the fairness of predictive student models through slicing analysis. In *Proceedings of the 9th international conference on learning analytics & knowledge* (pp. 225-234).
- [9] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y., 2014. Generative adversarial nets. *Advances in neural information processing systems*, 27.
- [10] Hakami, E. and Hernández Leo, D., 2020. How are learning analytics considering the societal values of fairness, accountability, transparency and human well-being?: A literature review. In: Martínez-Monés A, Álvarez A, Caeiro-Rodríguez M, Dimitriadis Y, editors. *LASI-SPAIN 2020: Learning Analytics Summer Institute Spain 2020: Learning Analytics. Time for Adoption?*; 2020 Jun 15-16; Valladolid, Spain. Aachen: CEUR; 2020. p. 121-41.
- [11] Hardt, M., Price, E., & Srebro, N. (2016). Equality of opportunity in supervised learning. *Advances in neural information processing systems*, 29.
- [12] Holsapple, C.W. and Lee-Post, A., 2006. Defining, assessing, and promoting e-learning success: An information systems perspective. *Decision sciences journal of innovative education*, 4(1), pp.67-85.
- [13] Hu, L.T. and Bentler, P.M., 1999. Cutoff criteria for fit indexes in covariance structure analysis: Conventional criteria versus new alternatives. *Structural equation modeling: a multidisciplinary journal*, 6(1), pp.1-55.
- [14] Knight, S., Friend Wise, A., and Chen, B. 2017. Time for Change: Why Learning Analytics Needs Temporal Analysis. *Journal of Learning Analytics*, 4(3), 7–17.
- [15] Kokoç, M., Akçapınar, G., and Hasnine, M. N. 2021. Unfolding Students' Online Assignment Submission Behavioral Patterns using Temporal Learning Analytics. *Educational Technology & Society*, 24(1), 223–235. <https://www.jstor.org/stable/26977869>
- [16] Larrabee Sønderlund, A., Hughes, E. and Smith, J. 2019. The efficacy of learning analytics interventions in higher education: A systematic review. *Br J Educ Technol*, 50: 2594-2618. <https://doi.org/10.1111/bjet.12720>
- [17] Leitner P., Khalil M., and Ebner M. 2017. Learning Analytics in Higher Education—A Literature Review. In: Peña-Ayala A. (eds) *Learning Analytics: Fundamentals, Applications, and Trends. Studies in Systems, Decision and Control*, vol 94. Springer, Cham. https://doi.org/10.1007/978-3-319-52977-6_1
- [18] Lundberg, S.M., Erion, G., Chen, H., DeGrave, A., Prutkin, J.M., Nair, B., Katz, R., Himmelfarb, J., Bansal, N. and Lee, S.I., 2020. From local explanations to global understanding with explainable AI for trees. *Nature machine intelligence*, 2(1), pp.56-67.
- [19] Maatuk, A.M., Elberkawi, E.K., Aljawarneh, S. et al. 2021. The COVID-19 pandemic and E-learning: challenges and opportunities from the perspective of students and instructors. *J Comput High Educ* (2021). <https://doi.org/10.1007/s12528-021-09274-2>
- [20] Mubarak, AA, Cao, H, Zhang, W, Zhang, W. Visual analytics of video-clickstream data and prediction of learners' performance using deep learning models in MOOCs' courses. *Comput Appl Eng Educ*. 2021; 29: 710– 732. <https://doi.org/10.1002/cae.22328>
- [21] Parkes, M., Stein, S., and Reading, C. 2015. Student preparedness for university e-learning environments. *The Internet and Higher Education*, 25: 1–10.
- [22] Prensky, M. 2001. Digital natives, digital immigrants Part 1. *On the Horizon*, 9(5), 1-6.
- [23] Riazy, S., Simbeck, K. and Schreck, V., 2020. Fairness in Learning Analytics: Student At-risk Prediction in Virtual Learning Environments. In *CSEU* (1) (pp. 15-25).
- [24] Smith, P. J., Murphy, K. L., and Mahoney, S. E. 2003. Towards identifying factors underlying readiness for online learning: An exploratory study. *Distance Education*, 24(1), 57–67. <https://doi.org/10.1080/01587910303043>
- [25] Tempelaar, D. 2020. Supporting the less-adaptive student: the role of learning analytics, formative assessment and blended learning. *Assessment & Evaluation in Higher Education*, 45:4, 579-593, DOI: 10.1080/02602938.2019.1677855
- [26] Thomas, P.S., Castro da Silva, B., Barto, A.G., Giguere, S., Brun, Y. and Brunskill, E., 2019. Preventing undesirable behavior of intelligent machines. *Science*, 366(6468), pp.999-1004.
- [27] Umer, R., Susnjak, T., Mathrani, A. and Suriadi, S., 2017. On predicting academic performance with process mining in learning analytics. *Journal of Research in Innovative Teaching & Learning*.

[28] Watkins, R., Leigh, D., and Triner, D. 2004. Assessing readiness for e-learning. *Performance Improvement Quarterly*, 17(4), 66-79.

[29] Yusuf, B.N. and Ahmad, J., 2020. Are we prepared enough? A case study of challenges in online learning in a private higher learning institution during the Covid-19 outbreaks. *Advances in Social Sciences Research Journal*, 7(5), pp.205-21

APPENDIX

Table 1. E-learning preparedness survey questionnaire

Questions	Response
(Q1) Does your device have a webcam and/or a microphone that will work with Zoom for virtual office hours and Q&A sessions?"	1 - Both webcam and microphone 2 - Only microphone
(Q2) What type of device will you mainly use to access [online learning module] for [course name]?"	1 - Desktop while on campus, plugin-bound laptop at home 2 - Laptop 3 - Desktop computer 4 - Tablet 5 - Chromebook
(Q3) "Are you currently living in [city] with the same time zone?"	1 - Yes 2 - No
(Q4) "How prepared do you feel for online learning this semester?"	1 (Not prepared at all) to 6 (Very prepared)

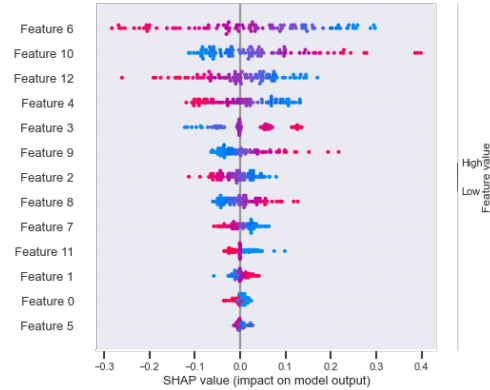
Table 2. Course content familiarity survey questionnaire

Questions	Response
(Q5.1) Please rate your level of confidence in the following assessment-related tasks: Creating new exam questions	1- Not confident at all
(Q5.2) Creating authentic assessments	
(Q5.3) Creating scoring tools (e.g., rubrics)	2- Slightly confidence
(Q5.4) Creating digital assessments	
(Q5.6) Using a variety of assessment methods	3- Fairly confident
(Q5.7) Fairness of your marking and grading	
(Q5.8) Consistency of your marking/grading	4- Completely confident
(Q5.9) Speed of your marking and grading	
(Q6) Which of the assessment-related concepts are you familiar with?	

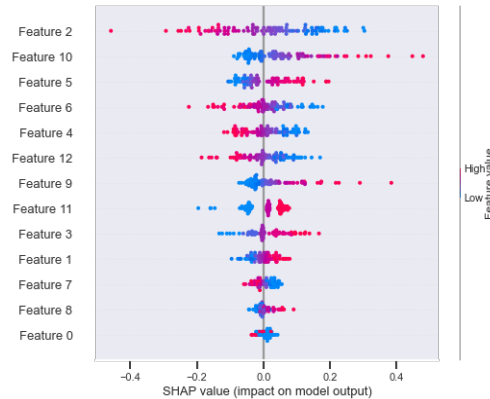
Table 3. Seldonian Algorithm and Logistic Regression Model Performance

Performance score quantile	Odd(0.015)		Odd(0.025)		Odd(0.05)		LR	
	ACC.	F1	ACC.	F1	ACC.	F1	ACC.	F1
y>quantile(.10)	0.43	0.57	0.59	0.74	0.58	0.71	0.91	0.95
y>quantile(.20)	0.56	0.68	0.62	0.73	0.62	0.73	0.84	0.91
y>quantile(.30)	0.42	0.50	0.42	0.49	0.33	0.54	0.78	0.86
y>quantile(.40)	0.32	0.33	0.39	0.46	0.72	0.76	0.76	0.80
y>quantile(.50)	0.36	0.34	0.36	0.32	0.37	0.34	0.73	0.73
y>quantile(.60)	0.44	0.29	0.52	0.52	0.42	0.22	0.75	0.67
y>quantile(.70)	0.54	0.43	0.47	0.18	0.49	0.16	0.79	0.60
y>quantile(.80)	0.57	0.09	0.42	0.35	0.44	0.27	0.84	0.49
y>quantile(.90)	0.57	0.16	0.36	0.20	0.35	0.17	0.90	0.30

Unconstrained Model Features (Biased)



Constrained Model Features (Bias-reduced)



- Feature 0 Sequence frequency: [Assignment activities]
- Feature 1 Skewness of cumulative counts of the number of logs
- Feature 2 Cumulative mean of number of unique activities participated
- Feature 3 Percentile of number of unique activities
- Feature 4 Assignment 1 Submission Rank
- Feature 5 Quiz 2 Submission Rank
- Feature 6 Quiz submission average time difference
- Feature 7 Week 2 learning activities completion time difference
- Feature 8 Week 4 learning activities completion time difference
- Feature 9 Sequence frequency: [Lecture => Watch Video]
- Feature 10 Sequence frequency: [Learning Resources]
- Feature 11 Cumulative max of the number of unique events
- Feature 12 Entropy of overall learning activity participation

Figure 3. Final model weights comparison

Leveraging Auxiliary Data from Similar Problems to Improve Automatic Open Response Scoring

Raysa Rivera-Bergollo
Worcester Polytechnic Institute
rriverabergollo@wpi.edu

Sami Baral
Worcester Polytechnic Institute
sbaral@wpi.edu

Anthony Botelho
University of Florida
abotelho@coe.ufl.edu

Neil Heffernan
Worcester Polytechnic Institute
nth@wpi.edu

ABSTRACT

As computer-based learning platforms have become ubiquitous, there is a growing need to better support teachers. Particularly in mathematics, teachers often rely on open-ended questions to assess students' understanding. While prior works focusing on the development of automated open-ended work assessments have demonstrated their potential, many of those methods require large amounts of student data to make reliable estimates. We explore whether a problem specific automated scoring model could benefit from auxiliary data collected from similar problems to address this "cold start" problem. We examine factors such as sample size and the magnitude of similarity of utilized problem data. We find the use of data from similar problems not only provides benefits to improve predictive performance by increasing sample size, but also leads to greater overall model performance than using data solely from the original problem when sample size is held constant.

Keywords

Auto-scoring, Open-ended questions, Mathematics, Natural Language Processing, Machine Learning

1. INTRODUCTION

The development of online learning platforms [12, 10] have transformed the instructional practices and learning experiences in traditional and expanded learning environments. These online-based learning platforms offer automated supports for assessing students' work as well as providing feedback. While in the past these supports were generally restricted to closed-ended problems with a finite number of accepted correct responses, advancements in machine learning and natural language processing methods have led to the development of tools that support open-ended work [15, 4, 2]. As open-ended questions in mathematics are widely used by teachers to understand the students' knowledge state and

their understanding of a topic, these types of tools have great utility for both teachers and students using these systems.

Automatically scoring mathematical expressions and explanations has several distinctive challenges due to the interleaving of linguistic and non-linguistic terms (e.g. such as numbers and mathematical expressions). For example, [13] provides automatic grading and feedback for math open response questions using clustering techniques, but it ignores all text explanations to focus solely on numerical expressions. In the past few years, there have been several works focused on the development and improvement of automated methods for assessing student open-ended responses in mathematics [6, 18, 17, 8]. These methods are mostly based on evaluating given student answers based on historical student answers and the scores given by teachers to such data. [6] compared the performance of different models for scoring math open-ended responses and attempted to establish a benchmark evaluation procedure to evaluate future models. Building on that work, [2] improved performance by using Sentence-BERT (SBERT) [14] embeddings on the same dataset to score student responses. SBERT modifies the pre-trained BERT (Bidirectional Encoder Representations from Transformers) [5] model to generate sentence-level embeddings. Similar approaches are utilized in recommending feedback messages for teachers to give to students.

As is prevalent in several machine learning applications, many of these approaches are susceptible to the cold start problem, where implementations of such methods may lack sufficient data to make informed estimates. While the impact will vary depending on the model and the context, most assessment models require non-trivial amounts of data to make accurate predictions (c.f. [1]) which may take time and effort to acquire. In cases when there is a newer student response that has not been encountered in the past, these types of methods often fall behind in suggesting an accurate score/feedback message posing this as the cold start problem. In light of this, mitigating the impact of this cold start problem would provide support for teachers across a wider range of problems. Transfer learning [16] is commonly used as a means of addressing the cold start problem. Within the field of mathematics education, we may be able to leverage data from similar content to improve performance in cases where there would otherwise be insufficient data to train an automated assessment model.

R. Rivera-Bergollo, S. Baral, A. Botelho, and N. Heffernan. Leveraging auxiliary data from similar problems to improve automatic open response scoring. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 679–683, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853119>

We seek to explore the effectiveness of leveraging auxiliary data (student responses) to similar open-ended problems in the auto-scoring of a new problem with limited labeled data. With the goal of addressing the cold start problem, we intend to answer the following research questions: 1. Does the addition of new labeled data from a similar open-response problem, improve the predictive performance of single problem based auto-scoring models? 2. Does leveraging data from a similar problem lead to better model performance in comparison to using data from a randomly selected problem? 3. What is the effect of incorporating auxiliary data into the training of an auto-scoring model and are there any benefits beyond that of increasing sample size?

2. METHODOLOGY

2.1 Dataset

For this study, data¹ consists of all student answers that have ever been submitted to open-ended problems within ASSISTments. For this study, we arbitrarily selected an open response problem within this dataset that contained at least 40 student responses ($n=45$) to act as a representative problem. For consistency of terminology, this representative problem will be referred to as the “original problem” throughout this paper, and will represent the problem for which we would like to train an auto-scoring model (e.g. we will treat it as the problem with insufficient data).

The selected problem pertains to logarithms, and presents the students with the following equation: “ $5\log(x+4) = 10$ ”; students are asked to either solve for x and explain their steps to solve or to type “no solution” if no viable solution exists. We then collaborated with a content expert to select a similar open-ended problem with a comparable number of existing labeled student answers ($n=43$) to train a model. This second problem, referred to as the “similar problem” throughout the remainder of this paper, had the same prompt as the original problem but with the following equation: “ $\log_2(1-x) = 4$ ”. While we acknowledge that the selected problems border on the threshold of what might be considered open-ended, much of the content of open curricula pair close-ended and open-ended components within many of their questions (e.g. solve and explain). In this way, the selected problems result in sufficient variation in student answers to examine auto-scoring models, and allow us to easily identify a problem with undeniable similarity both in terms of content and structure. As part of our analyses, we removed any problem from the remaining dataset containing fewer than 10 labeled student responses to mimic a practical application where such problems would not be considered sufficient in providing auxiliary data since we will be sampling from random problems.

Minor preprocessing was performed on the data to match the format of [2] which introduced the SBERT-Canberra model. These steps included the removal of HTML tags, other special characters, and references to images. Like in [6, 2], teacher-provided scores follow a 5-point integer scale

¹The data and code used in this work cannot be publicly posted due to the potential existence of personally identifiable information contained within student open response answers. In support of open science, this may be shareable through an IRB approval process. Inquiries should be directed to the trailing author of this work.

ranging from 0, indicating poor performance, through 4, indicating high performance. While we acknowledge that ordinal relationships are lost by representing the labels in this way, the scale is converted to a 5-valued categorical one-hot encoded vector and modeled as a multi-class prediction task (i.e. the model treats each score as a mutually-exclusive label) to keep consistent with [2].

2.2 Model

The “SBERT-Canberra” model [2] follows a similarity-ranking procedure to generate its predictions. When producing a prediction for a given student response, it applies SBERT to generate an embedding that captures semantic and syntactic meaning, such that similar responses are mapped to closer points within the embedding space. The SBERT embedding for this student response is compared to SBERT embeddings of historic labeled student responses. Using the Canberra distance measure [11], the score for the historic response corresponding to the smallest distance (i.e. the most similar response) is used as the score prediction. We chose to use this model as 1) it outperformed existing benchmarks in assessing student responses in mathematics [2], 2) no “training” involved in the traditional machine learning sense so we do not need to optimize hyperparameters, and 3) the model performance is directly linked to the scale and diversity of the historic responses.

2.3 Model Evaluation

To examine the use of auxiliary data, we conduct 2 analyses that each compare the SBERT-Canberra model with 3 different training sets. The analyses follow a bootstrapping procedure which samples with-replacement from the available data at increasing intervals. At each interval, student responses are randomly sampled to train and evaluate the model using 10-fold cross validation, where sampling is conducted within the training folds. This entire process is repeated 25 times, with the model performance being averaged across these iterations (to reduce noise caused by unlucky sampling). To evaluate the scoring results, the area under the curve, AUC, (calculated using the simplified multi-class calculation of ROC AUC from [9]) is used as the primary metric to compare the model’s predicted score of a student response to the actual score that was provided by a teacher.

The models are distinguished by their training data. The *Baseline Model* uses only student responses from the original problem. The *Similar Problem Model* uses a combination of student responses from the original problem as well as auxiliary responses sampled from the similar problem. Finally, the *Random Problem Model* uses a combination of student responses from the original problem as well as student responses sampled from 5 randomly-selected problems from the remaining dataset; per design and due to the scale of the data used, it is very unlikely for these problems to be similar to the original problem, allowing for comparisons to be made in regard to differing magnitudes of similarity.

We randomly sample 40 scored responses from the similar problem and from the 5 random problems to create a comparable set. Due to the large variations in sample sizes across problems within the dataset, we sample student responses for the Random Problem Model using a stratified selection method. From the 5 randomly-selected problems per inter-

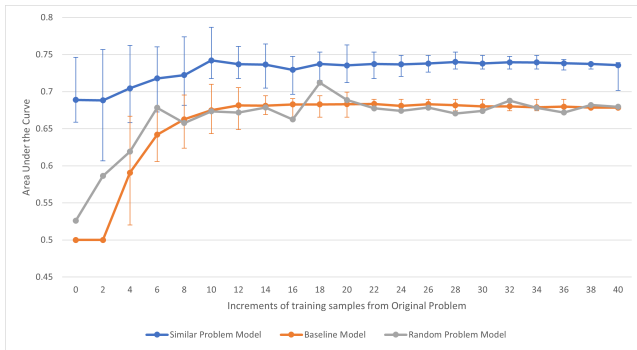


Figure 1: Average AUC varying original problem sample size.

val, 8 scored student responses are randomly selected per iteration in the interval and they compose the 40 samples to supplement the training data from the original problem.

The first analysis replicates a real-world scenario where we may have a small number of labeled samples for the original problem, but a larger number of samples that may be leveraged from other problems. For each bootstrapping interval, we randomly sample data from the original problem ranging from 0 to 40. The average performance of each model is then plotted with 95% confidence intervals calculated over the 25 repeated runs per interval. While the Baseline model is limited to only the 0 to 40 original problem samples, both the Similar Problem Model and Random Problem Model are able to use 40-80 samples over the set of intervals.

As it is hypothesized that the largest benefit of using auxiliary data is the added sample size, we conduct a second bootstrapping analysis that observes a constant sample size while varying the proportion of data used from the original problem. All models (except for the baseline) utilize 40 samples allowing us to see how the source of content affects model performance independent of data scale. The percentage intervals range from 0% to 100% of the training samples are from the original problem in 10% increments. So, at the first interval, all samples are responses from other problems, while at the end, all 40 samples are from the original problem. As the Baseline Model only utilizes data from the original problem, we are unable to maintain a consistent sample size across intervals. For comparative purposes, we increase the training sample size with the increasing percentage (i.e. using 0 samples, then 4 corresponding with 10%, etc.).

3. RESULTS AND DISCUSSION

For intervals 0 and 0%, no training data was provided for the baseline model so the average AUC of the baseline model is assigned to be 0.5 which is equivalent to chance.

Observing the Similar Problem Model in Figure 1, the model outperforms the average AUC of the baseline model across every increment of training samples from the original problem by approximately 0.073 in terms of average AUC per interval. This difference is also statistically reliable across a majority of intervals by comparing the confidence intervals.

Regarding the Random Problem Model, the model outper-

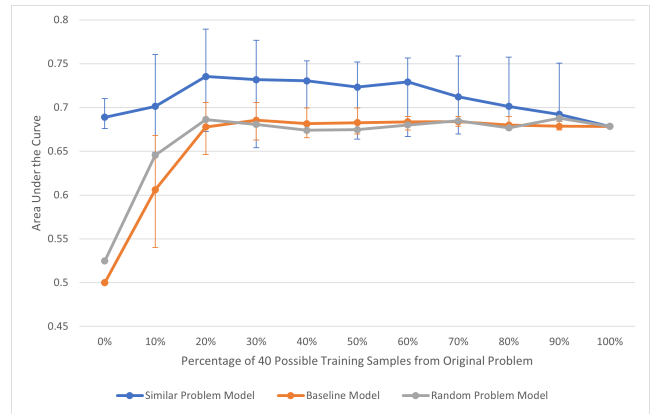


Figure 2: Average AUC varying sample proportion.

forms the average AUC of the baseline model across 43% of the increments tested. At an average difference of just 0.007 in terms of average AUC per interval, very little difference is observed between the Random Problem Model and the Baseline Model. It is worth noting that the performance of the Random Problem Model does outperform the Baseline over the initial intervals when sample size is the smallest, suggesting that even randomly-selected problems may provide benefit. However, this model also exhibited large variations in performance, leading us to omit the error bars to improve the readability of the figure; this variation is presumably attributable to the random selection of problems with varying magnitudes of similarity to the original problem.

An interesting trend emerged in regard to the Similar Problem Model as seen in Figure 2. When using 40 total training samples (and keeping this constant) with some percentage of samples from the original problem and the remaining samples from the similar problem, the modified model outperforms or equals the average AUC of the baseline model across every increment of training samples from the original problem by around 0.053 in terms of average AUC per interval. After the peak performance in terms of average AUC, the model's performance lessens as the percentage of training samples coming from the original problem increases.

The Random Problem Model follows closely with the performance of the baseline. When using 40 total training samples with some percentage of samples from the original problem and the remaining samples from 5 random problems, the modified model outperforms or equals the average AUC of the baseline model across 54% of the increments tested and by around 0.005 in terms of average AUC per interval.

The Baseline Model across both analyses provide insights into the current implementation of auto-scoring models. While the performance of the SBERT-Canberra model will likely vary across problems, we observe here that the model converges within a relatively small set of samples. After training from 12 samples from the original problem, the baseline model converges in terms of average AUC performance. It does seem to matter, however, which samples are used to train the model. We can see in both analyses that the Baseline Model's confidence intervals decrease with more sam-

ples. The relatively wide bounds over low sample sizes suggests that there are subsets of training samples that are better than others. This is not surprising as the diversity of data is often considered just as important as the scale in many machine learning applications [7].

There is a similar trend in regard to the scale of confidence bounds for the Similar Problem Model. Although the average AUC performance stabilized after 10 samples, the confidence intervals continued to shrink in the first analysis, but remained relatively constant in the second analysis. In both analyses, however, we see consistent, if not statistically reliable differences in comparison to the Baseline Model. In addressing our first research question, this finding suggests that the use of auxiliary data can lead to notable benefits to model performance. We see in the first analysis that the added sample size leads to notable performance through all intervals. While our initial hypothesis was that this benefit would likely be attributable to increased sample sizes, the trend of this Similar Problem Model in the second analysis contradicts that hypothesis. While this model still outperforms the baseline, as sample size is held constant, this cannot be the contributing factor to the differences we observe. We expected the final interval of Figure 2 to be an upper bound for model performance as this is when the data is most closely related to the test set, but we found that the inclusion of data from a similar problem added benefits that extend beyond the impact of sample size. This finding addresses our third research question, but still remains inconclusive as to what benefit is provided. It is possible, for example, that the auxiliary data acts as a regularization method (c.f. [3]), but the analyses conducted here are only able to rule out sample size being the contributing factor. These findings further confirm that scoring models can be improved upon when provided with more varied training samples from both the problem it is trying to score and similar problems rather than only being trained from samples of the original problem. Even when trained with the same number of samples, the Similar Problem model’s average AUC decreases after a peak training percentage composition which supports the theory that the quality of the training samples from the original problem are less than the quality of the combined samples.

What is perhaps most surprising about this comparison in the second analysis is that the model trained from 100% of data from the similar problem seems to outperform the model trained from 100% of the original problem. We believe that this is an artifact of the selected problems and the level of similarity that they exhibit. As such, we would not expect this finding to extend to every open-ended problem, but rather could extend to a subset where there is strong similarity between problems both in terms of content and the structure of student responses; this is the scenario where we believe this method would provide the most benefit.

This is particularly the case considering that the same level of benefit was not observed in regard to the Random Problem Model across the two analyses. Our hypothesis, as previously introduced, is that the added benefit is likely correlated with the magnitude of problem similarity. Even if this hypothesis is flawed, we are seeing that certain subsets of problems lead to better performance than others, emphasizing

the importance in selecting suitable problems from which to draw auxiliary data. In light of this, we can address our second research question in that problem similarity, loosely defined, does seem to impact performance.

4. CONCLUSIONS AND FUTURE WORK

In this paper, we explore a possible solution to the cold-start problem in automating the assessment of student open-ended work. We have shown that our SBERT-Canberra method using similar auxiliary problem data consistently and significantly outperformed the model using data solely from the original problem. When there are few training samples, even the modified SBERT-Canberra method using random problems’ data to supplement helped improve the performance. Throughout the exploration of both analyses, there is a noticeable benefit to supplementing the training samples with data from other problems. By supplementing the original training samples with multiple similar problems, we hypothesize that it will lead to even larger performance improvements to automatic scoring regardless of the number of original training samples. This would be particularly the case if our hypothesis is correct where some of this benefit is derived from regularizing factors.

The largest limitation is that this paper focuses on predicting the scores of only one specific problem. While we argue that the analyses conducted here were sufficient to address our research questions, there is a larger uncertainty that remains in regard to how representative these results are. This work should be tested across a variety of problems to ensure that the results generalize well to other problems. When deciding what constitutes a similar problem, future work could explore other methods that consider a wide range of comparison characteristics. Descriptives including the problem text, knowledge component, grade level, average difficulty, etc may be utilized in comparing problems to determine similarity. Defining such attributes would also provide opportunities to build models to better understand how matching characteristics correlate with model performance gains.

Future work should use transfer learning to use the SBERT-Canberra model of a similar problem as a starting point to score a new problem’s open-ended response. As more data from problems are collected, we found that there may still be benefits to using auxiliary data even beyond addressing the cold start problem. Furthermore, teachers often need supports in providing more meaningful feedback beyond that of a numeric score. ASSISTments is already able to recommend feedback for trained problem models, but it requires a lot of data in order to do so (more than for the automated scoring task). The use of auxiliary data as explored in this work may prove useful in other such contexts.

5. ACKNOWLEDGMENTS

We thank multiple NSF grants (e.g., 1917808, 1931523, 1940236, 1917713, 1903304, 1822830, 1759229, 1724889, 1636782, 1535428, 1440753, 1316736, 1252297, 1109483, & DRL-1031398); IES R305A170137, R305A170243, R305A180401, R305A120125, R305A180401, & R305C100024, the GAANN program (e.g., P200A180088 & P200A150306), and the EIR, ONR (N00014-18-1-2768), Schmidt Futures and a second anonymous philanthropy.

6. REFERENCES

- [1] M. Banko and E. Brill. Scaling to very very large corpora for natural language disambiguation. In *Proceedings of the 39th annual meeting of the Association for Computational Linguistics*, pages 26–33, 2001.
- [2] S. Baral, A. Botelho, J. Erickson, P. Benachamardi, and N. Heffernan. Improving automated scoring of student open responses in mathematics. In *Proceedings of the Fourteenth International Conference on Educational Data Mining, Paris, France, 2021*.
- [3] J. Bouwman. Quality of regularization methods. *DEOS Report 98.2*, 1998.
- [4] S. A. Crossley, K. Kyle, and D. S. McNamara. The tool for the automatic analysis of text cohesion (taaco): Automatic assessment of local, global, and text cohesion. *Behavior research methods*, 48(4):1227–1237, 2016.
- [5] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.
- [6] J. A. Erickson, A. F. Botelho, S. McAteer, A. Varatharaj, and N. T. Heffernan. The automated grading of student open responses in mathematics. In *Proceedings of the Tenth International Conference on Learning Analytics & Knowledge*, pages 615–624, 2020.
- [7] H. J. Hadi, A. H. Shnain, S. Hadishaheed, and A. Ahmad. Big data and five v’s characteristics. 2014.
- [8] G. Haldeman, M. Babeş-Vroman, A. Tjang, and T. D. Nguyen. Csf: Formative feedback in autograding. *ACM Transactions on Computing Education (TOCE)*, 21(3):1–30, 2021.
- [9] D. J. Hand and R. J. Till. A simple generalisation of the area under the roc curve for multiple class classification problems. *Machine Learning*, 45:171–186, 2001.
- [10] N. T. Heffernan and C. L. Heffernan. The assistments ecosystem: Building a platform that brings scientists and teachers together for minimally invasive research on human learning and teaching. *International Journal of Artificial Intelligence in Education*, 24(4):470–497, 2014.
- [11] G. Jurman, S. Riccadonna, R. Visintainer, and C. Furlanello. Canberra distance on ranked lists. In *Proceedings of advances in ranking NIPS 09 workshop*, pages 22–27. Citeseer, 2009.
- [12] K. R. Koedinger, A. Corbett, et al. *Cognitive tutors: Technology bringing learning sciences to the classroom*. na, 2006.
- [13] A. S. Lan, D. Vats, A. E. Waters, and R. G. Baraniuk. Mathematical language processing: Automatic grading and feedback for open response mathematical questions, 2015.
- [14] N. Reimers and I. Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks, 2019.
- [15] R. D. Roscoe and D. S. McNamara. Writing pal: Feasibility of an intelligent writing strategy tutor in the high school classroom. *Journal of Educational Psychology*, 105(4):1010, 2013.
- [16] L. Torrey and J. Shavlik. Transfer learning. In *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, pages 242–264. IGI global, 2010.
- [17] X. Yang, L. Zhang, and S. Yu. Can short answers to open response questions be auto-graded without a grading rubric? In *International Conference on Artificial Intelligence in Education*, pages 594–597. Springer, 2017.
- [18] L. Zhang, Y. Huang, X. Yang, S. Yu, and F. Zhuang. An automatic short-answer grading model for semi-open-ended questions. *Interactive learning environments*, 30(1):177–190, 2022.

Modifying Deep Knowledge Tracing for Multi-step Problems

Qiao Zhang
Drexel University
Philadelphia, PA, USA
qiao.zhang@drexel.edu

Natasha Lalwani
Drexel University
Philadelphia, PA, USA
nl498@drexel.edu

Zeyu Chen
Drexel University
Philadelphia, PA, USA
ac4267@drexel.edu

Christopher J. MacLellan
Drexel University
Philadelphia, PA, USA
christopher.maclellan@drexel.edu

ABSTRACT

Previous studies suggest that Deep Knowledge Tracing (or DKT) has fundamental limitations that prevent it from supporting mastery learning on multi-step problems [15, 17]. Although DKT is quite accurate at predicting observed correctness in offline knowledge tracing settings, it often generates inconsistent predictions for knowledge components when used online. We believe this issue arises because DKT’s loss function does not evaluate predictions for skills and steps that do not have an observed ground truth value. To address this problem and enable DKT to better support online knowledge tracing, we propose the use of a novel loss function for training DKT. In addition to evaluating predictions that have ground truth observations, our new loss function also evaluates predictions for skills that do not have observations by using the ground truth label from the next observation of correctness for that skill. This approach ensures the model makes more consistent predictions for steps without observations, which are exactly the predictions that are needed to support mastery learning. We evaluated a DKT model that was trained using this updated loss by visualizing its predictions for a sample student learning sequence. Our analysis shows that the modified loss function produced improvements in the consistency of DKT model’s predictions.

Keywords

Deep knowledge tracing, loss function, online learning

1. INTRODUCTION

Intelligent tutoring systems are widely used in K-12 education and online learning platforms to enhance learning. Knowledge tracing algorithms are embedded in such intelligent tutoring systems to support automatic selection of the

problems a learner should work on next based on their mastery of different skills. There are multiple popular knowledge tracing algorithms that are frequently used to predict students’ performance in offline settings. While these approaches have all achieved satisfactory performance in these settings, there is only limited work investigating the use of knowledge tracing algorithms in online settings [11, 17].

Deep Knowledge Tracing (DKT) is a knowledge tracing approach that has gained in popularity in recent years. It employs a recurrent neural network (RNN) [16] to predict student’s correctness on problem-solving steps that use particular skills. Though some studies demonstrated that DKT outperforms other knowledge tracing models such as Bayesian Knowledge Tracing [1] and Performance Factors Analysis (PFA) [10], it has some fundamental limitations and drawbacks. For example, DKT’s neural network representation is not easily interpretable, making it difficult for people to understand DKT’s predictions. Additionally, Yeung and Yeung [15] identified two problems with DKT—the model fails to reconstruct the observed input, and the DKT predictions are inconsistent and fluctuate over time.

In this paper, we investigate the issue of inconsistent predictions. Our work explores the hypothesis that DKT’s inconsistent behavior is primarily due to its loss function. We propose a novel modification to the DKT loss functions designed to produce more consistent behavior. Multiple authors have proposed ways of modifying the loss function by adding regularization terms [7, 8, 15]. However, our research explores a novel modification that evaluates predictions for each skill that does not have an observed ground truth value by using the next observed correctness for that skill.

We use the “Fraction Addition and Multiplication, Blocked vs. Interleaved” dataset accessed via DataShop [5] to evaluate a DKT model generated through training with this new loss function by visualizing its predicted correctness for each skill at each time step in a heatmap. We then compare these results with the predictions generated by a DKT model trained using the original loss function. Our results indicate that training with the revised loss function produces a DKT model that generates more consistent predictions than one produced by training with the original loss function.

Q. Zhang, Z. Chen, N. Lalwani, and C. MacLellan. Modifying deep knowledge tracing for multi-step problems. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 684–688, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853145>

2. BACKGROUND

2.1 Knowledge Tracing

Knowledge tracing approaches model a student’s knowledge over time and predict their performance on future problem-solving steps. Knowledge tracing algorithms are embedded in Intelligent Tutoring Systems to support automatic selection of the next problem a student will practice [13]. Much of the research on knowledge tracing has explored its use in offline settings; however, little work has explored the use of knowledge tracing in online settings. In offline settings, knowledge tracing models are fit to existing data sets, typically to evaluate different knowledge component models to identify those that better fit the data. In contrast, the objective of online knowledge tracing is to keep track of the student’s level of mastery for each skill (or knowledge component) and/or predict the student’s future performance based on their past activity. In a nutshell, knowledge tracing seeks to observe, depict, and quantify a student’s knowledge state, such as the level of mastery of skills underlying the educational materials [6]. The outputs of knowledge tracing support mastery learning and intelligent selection of which problems a student should work on next.

2.2 Deep Knowledge Tracing

Pieche et al. [12] proposed the Deep Knowledge Tracing (DKT) approach, makes use of a Long Short-Term Memory (LSTM) [4] architecture (complex variant of Recurrent Neural Network, or RNN) to represent latent knowledge. The use of an LSTM has become increasingly popular because it reduces the effect of vanishing gradients. It employs cell states and three gates to determine how much information to remember from previous time-steps and also how to combine that memory with information from the current time-step.

The DKT model accept an input matrix X , which is constructed by one-hot encoding two pieces of information for each step: q_t , which represents the knowledge components, and a_t , which represents whether the question was answered correctly. The information at each time step is packed into a tuple denoted as $h_t = \{q_t, a_t\}$. h_0 represent the initial state at time 0 (where $t = 0$). The network outputs the prediction Y based on the input and previous state. Y is a matrix that represents the probability of each KC being correctly answered at each step by a given student. y_t is the predicted probability at time t .

The objective of DKT is to predict performance at the next iteration (given the data from time 0 to t , predict $t + 1$). To optimize next iteration results, a dot product of the output vector y_t and the one-hot encoded vector of the next practiced KC $\delta(q_{t+1})$ is calculated. We take the cross entropy (denoted as l) of the dot product, average over number of steps and number of students. All together, the original loss function of DKT $L_{Original}$ can be expressed as:

$$L_{Original} = \frac{1}{\sum_{i=1}^n (T_i - 1)} \sum_{i=1}^n \sum_{t=1}^{T_i-1} l(y_t \cdot \delta(q_{t+1}^i), a_{t+1}^i) \quad (1)$$

where n is the number of students, and T_i is the length of the interaction sequence for student i .

When the size of a dataset increases, deep knowledge tracing generally has an edge over the classical statistical models, such as Bayesian Knowledge Tracing, Streak Model or Performance Factor Analysis, when it comes to predicting learner performance. The original DKT work [12] demonstrated that it can produce tremendous gains in AUC (e.g., 25%) when compared to prior results obtained from other knowledge tracing models. However, subsequent work suggests that the gains are not as large as originally anticipated [14]. One of the key advantages of DKT over classical knowledge tracing methods, such as BKT, is that it has access to more precise information about the temporal order of interactions as well as information about KCs not involved in the current step [2]. We intend to leverage these advantages of DKT to support online knowledge tracing [17] and explore whether it is possible to get better mastery learning behavior when using DKT rather than classical knowledge tracing approaches, such as BKT.

2.3 Challenges with DKT

Even though DKT has many advantages over other knowledge tracing models like Bayesian Knowledge Tracing (BKT) [1], Streak Model [3] and Performance Factor Analysis (PFA) [10], the model still has several limitations. Specifically, DKT models are difficult to interpret [14], make inconsistent predictions [15], and only consider the correctness of skills that are observed on each time step [7].

Correctness		✗	✗	✗	✓	✓	✓	✗	✓
DKT Mastery	36%	0%	0%	100%	0%	0%	0%	2%	0%

Figure 1: This example, drawn from Zhang and MacLellan (2021) [17], shows DKT model predictions on a single knowledge component given one student correctness sequence.

Yeung and Yeung [15] identified that the DKT predictions are not consistent and fluctuate over time. They also showed that the DKT model fails to reconstruct the input information in its predictions. For example, DKT may predict lower correctness on steps tagged with a particular skill even when the student correctly performs steps that contain the skill. Figure 1 is an example of this effect. From the first to the third steps, the student did not answer the problem correctly, but DKT predicted the third step would have a 100% chance of being correct. From the fourth to the sixth steps, the student correctly answered the question while DKT’s predictions dropped. Upon closer investigation of the DKT model, we believe that this unexpected behavior is due to the way that the loss is computed.

Our previous work [17] highlighted DKT’s shortcoming with respect of giving reliable predictions of correctness on steps tagged with each skill during online knowledge tracing. We want to further investigate the issues that prevent DKT from giving consistent predictions in the scenario of multi-step problem solving and online knowledge tracing. In this paper, we propose a novel revision of DKT’s loss function. We will discuss our approach in Section 3.1.

3. METHODOLOGY

We propose a novel approach to make the DKT model predictions more consistent by modifying the loss function used during training. We trained and tested on the ‘‘Fraction Addition and Multiplication, Blocked vs. Interleaved’’ dataset accessed via DataShop [5] with 80% training data and 20% testing data. This data was collected from a study presented in [9], the students solved problems by interacting with a fraction arithmetic tutor and solved three different types of problems. The three problem types are: Add Different (AD), add fractions with different denominators; Add Same (AS), add fractions with same denominators; Multiplication (M), multiply two fractions.

We created two DKT models: one trained using the original DKT loss function and another trained using the modified loss function. We then used the two models to make predictions on the same student sequence. Lastly, we visualized the predictions for each knowledge component (KC) as heat maps and evaluated the prediction consistency by comparing the heat maps generated using the different DKT models.

All DKT models in this paper consists of a input layer, a hidden layer, and a output layer with size 28, 200, and 14, respectively. The number of knowledge components determines the size of the input and output layers. The LSTM (long short-term memory) contained 200 hidden units. We trained the model over 1000 epochs, with a learning rate of 0.0025, a dropout rate of 0.4, and a batch size of 5. The only difference between the original DKT approach and our approach is the loss function used during training.

3.1 Revision of DKT Loss Function

As outlined in Section 2.3, DKT’s original loss function only evaluates the DKT predictions that have observed ground truth values. To overcome this challenge, we propose a revision to the loss function. Rather than using the original ground truth values typically provided to DKT’s loss function, our revised approach uses modified ground truth data that fills in steps without any observations by taking the next observation of that skill (see Figure 2).

Skill 1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Skill 2	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Skill 3	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 2: Graphical depiction of \hat{a} . Colored cells denote observed student performance (0/red equals incorrect and 1/green equals correct). Cells with white backgrounds are extrapolated from the next observation of each skill.

Mathematically, we use \hat{a} to represent the updated ground truth values that populate missing cells using the value from the next observation of each skill, see Figure 2. For example, for a specific knowledge component, if there is no ground truth at t_i and the next ground truth is at t_{i+n} , then the \hat{a} contains an entry at t_i that has the same value as the entry at t_{i+n} . As a result, the entries from t_i to t_{i+n-1} would share the same ground truth with t_{i+n} .

Next, we updated the loss function so that it evaluates the

model’s predictions for all entries that have a value in the updated ground truth values (\hat{a}). Here is the mathematical representation of this new loss function:

$$L_{Next} = \frac{1}{\sum_{i=1}^n \sum_{k=1}^K (T_{i,k} - 1)} \sum_{i=1}^n \sum_{k=1}^K \sum_{t=1}^{T_{i,k}-1} l(y_{t,k}, \hat{a}_{t+1,k}^i) \quad (2)$$

This updated loss function will evaluate most of the DKT predictions that did not originally have observed ground truth values. Note, some predictions are still not evaluated (those that occur near the end and do not have a next observation to use for evaluation). Because this new loss function evaluates more of DKT’s predictions in between observations, we believe it will result in more stable predictions.

4. MODEL EVALUATION

To evaluate the performance of DKT model after revising the loss function, we took a complete student sequence and generated correctness predictions for each skill using the DKT model. We have 14 skills (knowledge components) and three types of problems as introduced in Section 3. There are 8 steps for an Add Different (AD) problem, 3 steps for an Add Same (AS) problem, and 3 steps for a Multiplication (M) problem. Figure 3 is a comparison of the student’s predicted mastery of each KC at each step when solving a problem (problem type shown on the x-axis). We use the color to represent DKT’s prediction, with green indicating the student mastering a skill and red indicating not mastering a skill. We use the numbers to represent the ground truth where 1 equals correct and 0 equals incorrect. Figure 3b shows a substantial improvement in prediction consistency over Figure 3a.

In Figure 3a, the DKT predictions fluctuate over time. There is also a pattern of inconsistent predictions on the ‘‘AD Right Convert Numerator’’, ‘‘AD Answer Numerator’’ and ‘‘AD Done’’ skill even though the ground truth values for these skills are 1 during the series of problems practiced. Initially, the DKT model trained using the original loss predicts that the student masters this skill after a few practices. However, we see that for certain repeating periods over the remainder of the sequence, the model predicts the student will get steps with these skills wrong. The student mastered these three skills initially. As the student starts solving additional steps, however, the DKT model alternates between correct and incorrect predictions over the remainder of the sequence. These behaviors are unexpected and contrary to the typical assumption that students will not forget skills once they obtain mastery.

In Figure 3b, the problem of wavy DKT predictions (alternating correct and incorrect predictions for different skills) is largely addressed. The DKT model with the revised loss predicts that the student obtains mastery on all the AD skills and retains this mastery through the end of training. The DKT predictions are consistent with the ground truth in this case.

These results suggest that our revised loss function produces more consistent DKT model predictions. Besides the

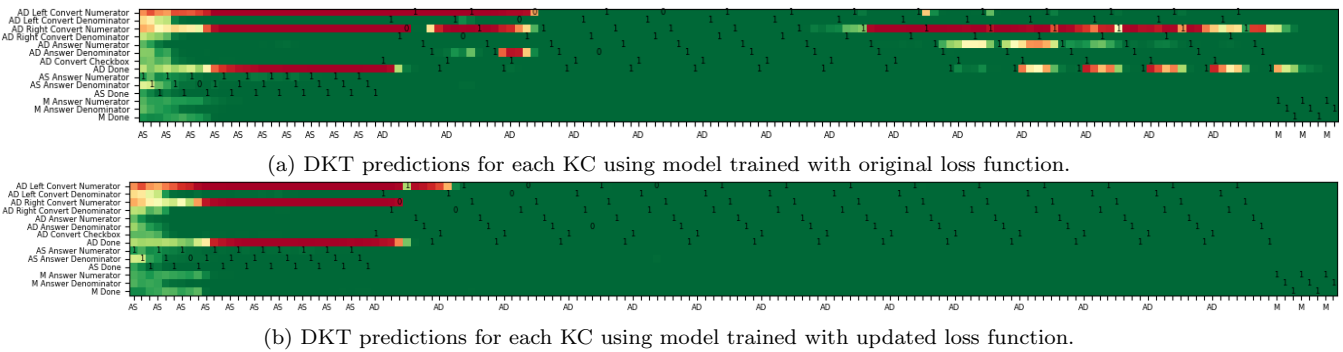


Figure 3: A comparison model performance between DKT models trained using the original and revised loss functions.

improvement, we noticed a common issue that occurred in both the original and the revised DKT model. The student started with 10 AS problems but both DKT models predict improvement of mastery in M and AD skills even before M and AD problems were given to the student. We believe that more work is needed to better understand how DKT relates the corresponding skills in a multi-step problem.

5. RELATED WORKS

Multiple authors have discussed the limitations of DKT in handling multi-skill sequences and possible modifications to the loss function to improve model behavior. Yeung and Yeung [15] proposed regularization terms to address the reconstruction problem (where model predictions move opposite to student performance) and the wavy prediction transition problem (where skill predictions cycle between high and low). Inspired by their study, we believe that revising the loss function is the key to enhancing the consistency of DKT model predictions. Rather than addressing these two problems separately using regularization terms, our approach modifies the loss function so that it evaluates predictions that lack ground truth observations.

Beyond modifying the loss function, Pan and Tezuka [8] proposed pre-training regularization, which incorporates prior knowledge by including synthetic sequences to the neural network before training DKT with real student data. Their motivation is similar to ours—their goal is also to solve the inverted prediction problem (referred to as the reconstruction problem by Yeung and Yeung). They added synthetic data to a baseline model trained with student data and then introduced two regularization measures to measure the severity of the inverted prediction problem. This approach is different from ours as we are using the ground truth value of each skill to populate skills and steps that do not have observations.

6. CONCLUSIONS AND FUTURE WORK

We revised DKT’s loss function to improve prediction consistency across all KCs over time. Our main contribution is that we propose a novel way of modifying the DKT loss function by evaluating skill predictions at the time steps that lack ground truth observations. Instead of only addressing DKT’s consistency issues, our ultimate goal is to use DKT as an approach to keep track of student performance in online learning environments and recommend problems to support personalized learning.

Through our heat map analysis, we demonstrated that a DKT model trained with our improved loss function generates more consistent predictions than a DKT model trained with the original loss. Our analysis showed that predictions for certain skills would cycle between high and low for a DKT model trained with the original loss function; i.e., generated inconsistent predictions over time. In contrast, the DKT model trained with the revised loss function showed much smoother, more consistent predictions that started lower and improved steadily over the course of training.

Moving forward, we have a number of additional future directions that we would like to explore to improve DKT’s stability and accuracy. In our current work, we propose an updated loss function that evaluates the DKT predictions for each skill in terms of the next observation of that skill. In future work, we instead want to evaluate each prediction in terms of all future predictions. Further, we plan to weight each evaluation by a decay factor γ as Yeung & Yeung [15] proposed in their future direction. Finally, we should move online and evaluate how well the revised DKT operates in an online mastery learning context.

7. ACKNOWLEDGMENTS

This work was funded by NSF award #2112532. The views, opinions, and/or findings expressed are those of the authors and should not be interpreted as representing the official view or policies of the funding agency.

8. REFERENCES

- [1] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4(4):253–278, 1994.
- [2] T. Gervet, K. Koedinger, J. Schneider, T. Mitchell, et al. When is deep learning the best approach to knowledge tracing? *Journal of Educational Data Mining*, 12(3):31–54, 2020.
- [3] N. T. Heffernan and C. L. Heffernan. The assistments ecosystem: Building a platform that brings scientists and teachers together for minimally invasive research on human learning and teaching. *International Journal of Artificial Intelligence in Education*, 24(4):470–497, 2014.
- [4] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

- [5] K. R. Koedinger, R. S. Baker, K. Cunningham, A. Skogsholm, B. Leber, and J. Stamper. A data repository for the edm community: The pslc datashop. *Handbook of educational data mining*, 43:43–56, 2010.
- [6] Q. Liu, S. Shen, Z. Huang, E. Chen, and Y. Zheng. A survey of knowledge tracing. *arXiv preprint arXiv:2105.15106*, 2021.
- [7] Q. Pan and T. Tezuka. Accuracy-aware deep knowledge tracing with knowledge state vector loss. pages 90–92, 2020.
- [8] Q. Pan and T. Tezuka. Prior knowledge on the dynamics of skill acquisition improves deep knowledge tracing. In *Proceedings of the 29th International Conference on Computers in Education*, pages 201–211, 2021.
- [9] R. Patel, R. Liu, and K. R. Koedinger. When to block versus interleave practice? evidence against teaching fraction addition before fraction multiplication. In *Proceedings of Cognitive Science Conference*, pages 2069–2074, 2016.
- [10] P. I. Pavlik, H. Cen, and K. R. Koedinger. Performance factors analysis—a new alternative to knowledge tracing. In *Proceedings of the 2009 conference on Artificial Intelligence in Education: Building Learning Systems that Care: From Knowledge Representation to Affective Modelling*, pages 531–538, 2009.
- [11] R. Pelánek. Bayesian knowledge tracing, logistic models, and beyond: an overview of learner modeling techniques. *User Modeling and User-Adapted Interaction*, 27(3):313–350, 2017.
- [12] C. Piech, J. Bassen, J. Huang, S. Ganguli, M. Sahami, L. J. Guibas, and J. Sohl-Dickstein. Deep knowledge tracing. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.
- [13] J. Rollinson and E. Brunskill. From predictive models to instructional policies. *Proceedings of the 8th International Conference on Educational Data Mining, EDM 2015*, pages 179–186, 2015.
- [14] X. Xiong, S. Zhao, E. G. Van Inwegen, and J. E. Beck. Going deeper with deep knowledge tracing. *International Educational Data Mining Society*, 2016.
- [15] C.-K. Yeung and D.-Y. Yeung. Addressing two problems in deep knowledge tracing via prediction-consistent regularization. In *Proceedings of the Fifth Annual ACM Conference on Learning at Scale*, pages 1–10, 2018.
- [16] W. Zaremba, I. Sutskever, and O. Vinyals. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*, 2014.
- [17] Q. Zhang and C. J. Maclellan. Going online: A simulated student approach for evaluating knowledge tracing in the context of mastery learning. *International Educational Data Mining Society*, 2021.

Clustering Students Using Pre-Midterm Behaviour Data and Predict Their Exam Performance

Huanyi Chen
University of Waterloo
huanyi.chen@uwaterloo.ca

Paul A.S. Ward
University of Waterloo
pasward@uwaterloo.ca

ABSTRACT

Student behaviour should correlate to the course performance. This paper explored different types of clustering algorithms using the pre-midterm student behaviour data. We found meaningful and interpretable results when clustering algorithms generate three clusters. The clusters can be briefly summarized as potential top performance (PTP) students, potential poor performance (PPP) students, and mixed performance (MP) students. We found that PTP students usually submit early and gain a high score, PPP students usually submit late and gain a low score, and MP students usually make most submissions. MP students are hard to cluster. However, we found a good connection between other students' behaviour and performance if we leave out MP students.

Keywords

Computer Education, Clustering, Student Behaviour, Auto-grading System, Student Performance

1. INTRODUCTION

Students expose different learning behaviours in programming courses. Several studies focused on analysing the students' data in order to understand student behaviours [5, 1, 14, 11]. The clustering technique was one of the common methodologies used in such studies. There are several benefits to identifying students' behaviours. First, groups of students with similar academic and behaviour characteristics would benefit from the same intervention, which can reduce the time for instructors to identify and implement the right intervention for individual students [12]. Second, a better understanding of students' misconceptions can lead to a better support system for novice programmers and provide adaptive feedback for students [4]. Last but not least, it is also a common technique used to predict student performance [6, 20, 17].

Although clustering techniques can be a valuable tool for re-

searchers to categorize different student behaviours, several studies only applied a single type of clustering algorithm in their experiments [4, 15, 3, 8]. However, clustering results can be affected by the wrong choice of the clustering algorithm, which can cause the result under the threat of validity. To address this issue, people should experiment with multiple clustering algorithms to confirm that different clustering algorithms produce similar results where the essential characteristics of the resulting clusters are identical. Our experiment confirms that clusters' essential characteristics are the same across multiple clustering algorithms, forming further discussion foundations.

Many studies applied clustering techniques to predict student performances [10, 7, 16]. However, the data used in these studies generally include both the pre-midterm data and the post-midterm data. Because of a strong correlation between the midterm exam grades and the final exam grades [2, 6, 9], which is also true for our data (Pearson correlation of 0.81), it suggests that students who failed the midterm are likely to fail the final exam. Therefore, it is critical to identify at-risk students before the midterm. Our study merged multiple clustering algorithms into a predictive model to predict students' performance using pre-midterm data.

This study applied the clustering techniques to students' behaviours exposed in their pre-midterm submissions in an auto-grading system, Marmoset [18], to categorize them into different clusters. We applied multiple clustering techniques and compared their results to remove any effects caused by the wrong choice of the clustering algorithm. Then we tried to predict students' performance using these clustering techniques. The research questions we want to ask are:

- **RQ1:** Are different clustering algorithms producing different results?
- **RQ2:** What are the characteristics of students in different clusters?
- **RQ3:** What were the exam grades of students in different clusters?

We want to clarify that we want to separate RQ1 and RQ2 because RQ1 is closely related to the predictive power of the clustering techniques, while RQ2 mainly focuses on providing better insights into the clusters.

H. Chen and P. Ward. Clustering students using pre-midterm behaviour data and predict their exam performance. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 689–694, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853143>

2. COURSE BACKGROUND

The data we used in the study is collected from an introduction-level programming course in an R1 university in Canada. Students were supposed to learn how to carry out operational tasks using the C and C++ languages, perform procedural and object-oriented programming, and other relevant programming knowledge. The preliminary experiments were based on the data of 130 students who attended both the midterm and final exams in the course. Programming knowledge was not required.

In the course, an auto-grading system, Marmoset [18], was used. We refer to a coding/programming question in Marmoset as a *task*. A task may contain multiple tests. For any task, a student can make multiple submissions against it. Marmoset will automatically test it for each submission and reveal some test results to the student. Students can thus learn some feedback from those test results and then improve or fix their code accordingly.

In the course, there were different types of coding questions.

i) Coding Labs: coding labs took place in the lab room. There was exactly one assigned task for each coding lab, which was to be completed and submitted during the lab period (2 hours in the morning). Before the midterm exam, the coding lab was scheduled weekly. After the midterm exam, the coding lab was scheduled biweekly. There was a corresponding extended deadline (the same date but in the evening). Some students may rely on that deadline rather than finish during the lab. **ii) Homework Assignments:** homework assignments were assigned for students to do at home. They were assigned during lab time. Before the midterm, homework problems were due the following week. After the midterm, homework problems were due approximately two weeks later. In every homework assignment, there were multiple tasks, all of which had the same deadline. **iii) Coding Examination:** there was an in-lab coding examination during the course. It was similar to a coding lab. However, its grade comprised a portion of the midterm grade. An extended deadline was also allowed for the coding examination.

3. EXPERIMENT AND RESULTS

3.1 Features

The auto-grading system Marmoset stored every student’s submission during the course. In our study, we extracted three features.

- **passrate:** for every Marmoset task, we calculated the best score (the best number of tests passed among the submissions) a student made before the task deadline. Then we divide the total number of tests of that task to form the *passrate* feature. Because every assignment had multiple tasks, so for assignments, we need to sum the tasks’ best scores to form the best score for an assignment, and we sum tasks’ total number of tests to form the total number of tests for an assignment. For example, assignment 1 had 4 tasks and a total number of 54 tests. If a student’s best submissions of that 4 tasks passed 6, 6, 7, and 8 tests separately, then the passrate of that student for assignment 1 will be $(6 + 6 + 7 + 8)/54 = 0.5$. This process was not

Table 1: Total number of tests for different assignments and coding labs pre-midterm, “a” stands for homework assignment, “l” stands for coding lab

assignment	# total tests	coding lab	# total tests
a1	54	l1	8
a2	85	l2	19
a3	73	l3	19
a4	87	l4	19

needed for coding labs since there was only one task in a coding lab. Table 1 shows the total number of tests for different assignments and coding labs.

- **lastsub:** we extracted the *lastsub* feature as how many minutes between the last submission a student made and the task deadline. If a student made any submission before the deadline, this feature would contain a non-zero value. Only if a student did not submit before the deadline can the value be zero. Note that this feature will be zero for students whose submissions met the extended deadline but did not meet the original deadline for coding labs.
- **nsub:** the *nsub* feature represents how many submissions a student made before the task deadline for a given task. For assignments, we summed the numbers of submissions of tasks to form the nsub of an assignment.

The features used in clustering algorithms is summarized in Table 2.

Table 2: Features, * ∈ {1, 2, 3, 4}, “a” stands for homework assignment, “l” stands for coding lab. Only pre-midterm data were used

feature name
a*_passrate
a*_lastsub
a*_nsub
l*_passrate
l*_lastsub
l*_nsub

3.2 RQ1: What are the clustering results from different clustering algorithms given the pre-midterm data?

For research question 1, we explored all types of the clustering algorithms provided in the scikit-learn python package [13]. It includes: *K-Means* (KM), *Affinity Propagation* (AP), *Spectral Clustering* (SC), *Hierarchical Clustering* (HC), and *Density-based Spatial Clustering* (DBSC). We standardize every feature by removing the mean and scaling to unit variance before clustering. For a sample x , the standard score is calculated as:

$$z = \frac{x - u}{s}$$

where u is the mean of the samples and s is the standard deviation of the samples.

We tested different options for setting the number of clusters in different algorithms. We found that setting the number to 3 will give us good interpretive results. In this section, we will only compare the labelling. For the characteristics of different clusters, we will discuss them in RQ2.

Table 3 presents the size of different clusters using different clustering algorithms. To compare the cluster results across different clustering algorithms, we used *adjusted rand index* [19] for evaluation. Random labelling samples will make the adjusted rand index close to 0.0, and the value will be exactly 1.0 when the clustering results are identical. Table 4 presents the results. Note that a cluster may be given different labels in different clustering algorithms. We re-numbered them according to the findings in the discussion of RQ2.

Table 3: The size of different clusters in different clustering algorithms

	KM	AP	HC	SC	DBSC
cluster 1	62	56	48	58	32
cluster 2	59	65	75	63	89
cluster 3	9	9	7	9	9

Table 4: The adjusted rand index results (high to low)

first algorithm	second algorithm	adjusted rand index
KM	AP	0.780665
AP	SC	0.730717
KM	SC	0.682500
KM	HC	0.557372
HC	DBSC	0.521486
HC	SC	0.515544
AP	HC	0.475727
AP	DBSC	0.414938
SC	DBSC	0.382402
KM	DBSC	0.322811

From Table 3 and Table 4, we can tell the clustering results from KM, AP, and SC are similar to each other (similar cluster sizes and high adjusted rand index value). In contrast, HC and DBSC produced different clustering results. Combining KM, HC and DBSC should help reduce the effect of an improper pick of the clustering algorithms.

3.3 RQ2: What are the characteristics of students in different clusters?

We carefully examined students in different clusters. Interestingly, although the cluster results differed in the cluster size and the adjusted rand index metric, we found that students in different clusters share similar characteristics across different clustering algorithms. We name the students in the three clusters as: **Potential-Top-Performance (PTP) students**, **Potential-Poor-Performance (PPP) students**, and **Mixed-Performance (MP) students**.

Table 5 summarizes the characteristics of different students.

We also examined the students that were put into different clusters from different algorithms. In general, those stu-

dents put into different clusters from different algorithms were those whose behaviour was in the middle of PTP students and MP students or the middle of PPP and MP students. However, we found there were no students put into the PTP cluster in one algorithm while being put into the PPP cluster in another algorithm. We can tell that students from these two clusters share no typical behaviour from any aspect.

3.4 RQ3: What were the exam grades of students in different clusters?

Because some students were put into different clusters from different clustering algorithms, we consider students in the PTP cluster only if they were put into the PTP cluster by all clustering algorithms. Similarly, students in the PPP cluster were only put into the PPP cluster by all clustering algorithms. The remaining students will be MP students.

In addition to the midterm grades and final exam grades, there was a coding examination grade, which will comprise a portion of the midterm grade. Figure 1 shows us the relation between the coding examination grades and different clusters. Figure 2 shows us the relation between midterm grades and final grades of different clusters.

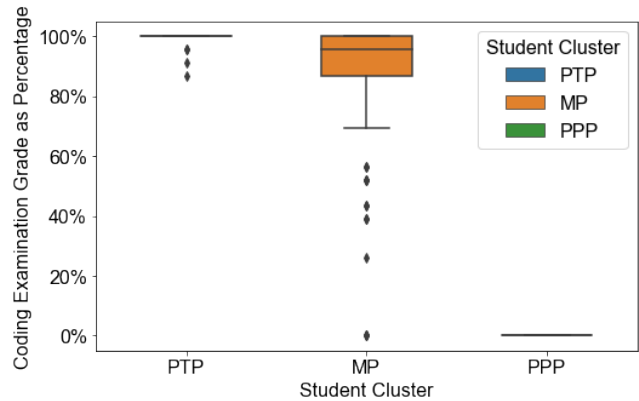


Figure 1: The box-plot of coding examination grades. PPP students all got zero while PTP students mostly got 100%.

From Figure 1, we can see PTP students achieved a very high score on the coding examination, while PPP students achieved 0% on the coding examination. The result is expected since PTP students mostly performed well on assignments and labs, which were programming questions. It is reasonable that they achieved a high score on the coding examination. In contrast, for PPP students, since they performed poorly on those questions, it is not surprising that they got a significantly low score.

From Figure 2, we can see the performances of MP students messed up with other clusters of students. However, if we exclude them, as shown in Figure 3, we can see the performances of PTP students and PPP students were completely different. The reason we set a cut off point as 50% for the exam grades is that it is the required grades for passing a course in the university.

It is essential to consider how many students genuinely need

Table 5: Characteristics of different clusters. Because there are multiple clustering algorithms, we calculated the medians of students of different clustering algorithms and then combined them into ranges. The lastsub feature of the coding lab 1 was treated separately from other labs because it was due on the second day rather than the 2-hour lab time, while lab 4 was due to the significant time differences observed.

	PTP students	MP students	PPP students
assignment passrate	96% – 98%	85% – 95%	34% – 66%
assignment lastsub	2 – 3 days early	16 – 27 hours early	2 – 6 hours
assignment nsub	25 – 29	26 – 42	8 – 21
lab 1-4 passrate	95% – 100%	lab 1,4: 100%, lab 2: 52% – 74%, lab 3: 0% – 53%	0%
lab 1 lastsub	14 – 16 hours early	3 – 4 hours early	7 – 14 minutes early
lab 2,3 lastsub	30 – 58 minutes early	0 – 15 minutes early	0 – 2 minutes early
lab 4 lastsub	70 – 77 minutes early	42 – 54 minutes early	0 – 2 minutes early
lab 1-4 nsub	3 – 6	lab 1,2,4: 4 – 6, lab 3: 1 – 2	0 – 4

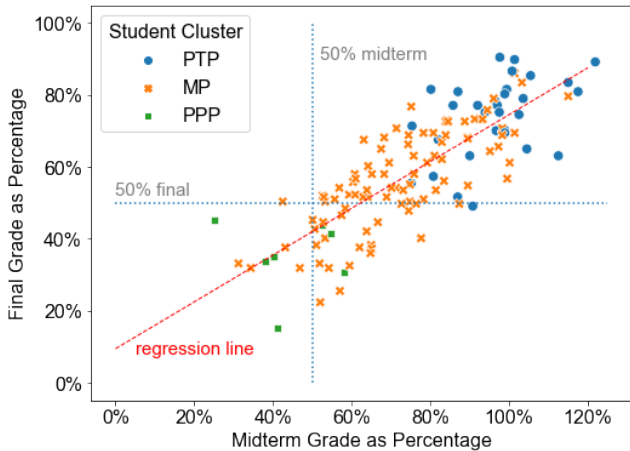


Figure 2: Midterm Grades and Final Exam Grades (as percentage). The total grade for midterm was 120% because there were bonus questions.

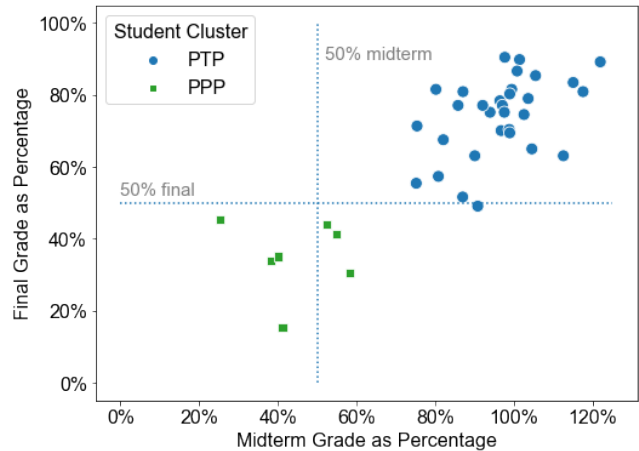


Figure 3: Midterm Grades and Final Exam Grades (as a percentage). The total grade for the midterm was 120% because there were bonus questions.

help from the identified PPP students. In other words, precision is important [2]. It is the higher, the better. Then we can calculate the precision of the PPP students, which is $4/7 = 0.57$ (4 students out of the total 7 PPP students were below 50% in the midterm). Similarly, the precision of the final grades is 1.0, since no PPP students had a final grade above 50%. These results are promising, implying that the clustering technique can predict student performance.

4. LIMITATIONS

The limitation of this study was that the data used in the study was of a limited amount. We appreciate any replicate studies to help validate the results in our study.

5. DISCUSSION

This study applied clustering techniques to pre-midterm students' behaviour data by using an auto-grading system, namely how early students make their last submissions, how many submissions they make, and the best score. We found that different clustering algorithms label students differently and put them into different clusters, thus providing different predictive power. However, combining k-means, hierarchical clustering, and density-based spatial clustering algorithms

should help reduce the negative effect of an improper pick.

We found that the clusters share the same characteristics regardless of the labelling. We can summarize those clusters as Potential Top Performance (PTP) cluster, Potential Poor Performance (PPP) cluster, and Mixed Performance (MP) cluster (in which students might be put into different clusters from different clustering algorithms). We found that students in PTP/PPP are generally exposing behaviours at the extreme, and they perform well/poorly on the exams.

Better understanding the MP students is one of the future works. For example, whether there are sub-groups within this cluster, how to understand their learning behaviours, and what factors might prevent them from being successful in the course?

Although we are still at the preliminary stage, we believe our finding allows a fair evaluation of the participation of students in a course. Also, because our finding shows that predicting at-risk students in advance is possible, thus corrective actions to improve the final results might be implemented to help people achieve a learning environment with fairness and equity.

References

- [1] M. An, H. Zhang, J. Savelka, S. Zhu, C. Bogart, and M. Sakr. Are Working Habits Different Between Well-Performing and at-Risk Students in Online Project-Based Courses? In *Proceedings of the 26th ACM Conference on Innovation and Technology in Computer Science Education V. 1*, pages 324–330, Virtual Event Germany, June 2021. ACM. ISBN 978-1-4503-8214-4. doi: 10.1145/3430665.3456320. URL <https://dl.acm.org/doi/10.1145/3430665.3456320>.
- [2] H. Chen and P. A. S. Ward. Predicting Student Performance Using Data from an Auto-Grading System. In *Proceedings of the 29th Annual International Conference on Computer Science and Software Engineering, CASCON '19*, pages 234–243, USA, 2019. IBM Corp. event-place: Toronto, Ontario, Canada.
- [3] R. W. Crues, G. M. Henricks, M. Perry, S. Bhat, C. J. Anderson, N. Shaik, and L. Angrave. How do Gender, Learning Goals, and Forum Participation Predict Persistence in a Computer Science MOOC? *ACM Transactions on Computing Education*, 18(4):1–14, Nov. 2018. ISSN 1946-6226. doi: 10.1145/3152892. URL <https://dl.acm.org/doi/10.1145/3152892>.
- [4] A. Emerson, A. Smith, F. J. Rodriguez, E. N. Wiebe, B. W. Mott, K. E. Boyer, and J. C. Lester. Cluster-Based Analysis of Novice Coding Misconceptions in Block-Based Programming. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education, SIGCSE '20*, pages 825–831, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 978-1-4503-6793-6. doi: 10.1145/3328778.3366924. URL <https://doi-org.proxy.lib.uwaterloo.ca/10.1145/3328778.3366924>. event-place: Portland, OR, USA.
- [5] S. C. Goldstein, H. Zhang, M. Sakr, H. An, and C. Dashti. Understanding How Work Habits influence Student Performance. In *Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education*, pages 154–160, Aberdeen Scotland Uk, July 2019. ACM. ISBN 978-1-4503-6895-7. doi: 10.1145/3304221.3319757. URL <https://dl.acm.org/doi/10.1145/3304221.3319757>.
- [6] A. Hellas, P. Ihanola, A. Petersen, V. V. Ajanovski, M. Gutica, T. Hynninen, A. Knutas, J. Leinonen, C. Messom, and S. N. Liao. Predicting academic performance: a systematic literature review. In *Proceedings Companion of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education*, pages 175–199, Larnaca Cyprus, July 2018. ACM. ISBN 978-1-4503-6223-8. doi: 10.1145/3293881.3295783. URL <https://dl.acm.org/doi/10.1145/3293881.3295783>.
- [7] J.-L. Hung, M. C. Wang, S. Wang, M. Abdelrasoul, Y. Li, and W. He. Identifying At-Risk Students for Early Interventions A Time-Series Clustering Approach. *IEEE Transactions on Emerging Topics in Computing*, 5(1):45–55, Jan. 2017. ISSN 2168-6750. doi: 10.1109/TETC.2015.2504239. URL <http://ieeexplore.ieee.org/document/7339455/>.
- [8] H. Khosravi and K. M. Cooper. Using Learning Analytics to Investigate Patterns of Performance and Engagement in Large Classes. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*, pages 309–314, Seattle Washington USA, Mar. 2017. ACM. ISBN 978-1-4503-4698-6. doi: 10.1145/3017680.3017711. URL <https://dl.acm.org/doi/10.1145/3017680.3017711>.
- [9] S. N. Liao, D. Zingaro, K. Thai, C. Alvarado, W. G. Griswold, and L. Porter. A Robust Machine Learning Technique to Predict Low-Performing Students. 19(3): 1–19. doi: 10.1145/3277569. URL <https://doi.org/10.1145/3277569>.
- [10] M. I. López, J. M. Luna, C. Romero, and S. Ventura. Classification via clustering for predicting final marks based on student participation in forums. page 4. URL <https://eric.ed.gov/?id=ED537221>.
- [11] K. Mierle, K. Laven, S. Roweis, and G. Wilson. Mining Student CVS Repositories for Performance Indicators. page 5, May 2005.
- [12] S. Mojarad, A. Essa, S. Mojarad, and R. S. Baker. Data-Driven Learner Profiling Based on Clustering Student Behaviors: Learning Consistency, Pace and Effort. In R. Nkambou, R. Azevedo, and J. Vassileva, editors, *Intelligent Tutoring Systems*, volume 10858, pages 130–139. Springer International Publishing, Cham, 2018. ISBN 978-3-319-91463-3 978-3-319-91464-0. doi: 10.1007/978-3-319-91464-0_13. URL http://link.springer.com/10.1007/978-3-319-91464-0_13. Series Title: Lecture Notes in Computer Science.
- [13] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and . Duchesnay. Scikit-Learn: Machine Learning in Python. 12: 2825–2830. ISSN 1532-4435.
- [14] V. Sher, M. Hatala, and D. Gaevi. Analyzing the consistency in within-activity learning patterns in blended learning. In *Proceedings of the Tenth International Conference on Learning Analytics & Knowledge*, pages 1–10, Frankfurt Germany, Mar. 2020. ACM. ISBN 978-1-4503-7712-6. doi: 10.1145/3375462.3375470. URL <https://dl.acm.org/doi/10.1145/3375462.3375470>.
- [15] D. B. Silva and C. N. Silla. Evaluation of students programming skills on a computer programming course with a hierarchical clustering algorithm. In *2020 IEEE Frontiers in Education Conference (FIE)*, pages 1–9, Uppsala, Sweden, Oct. 2020. IEEE. ISBN 978-1-72818-961-1. doi: 10.1109/FIE44824.2020.9274130. URL <https://ieeexplore.ieee.org/document/9274130/>.
- [16] S. Sisovic, M. Matetic, and M. B. Bakaric. Clustering of imbalanced moodle data for early alert of student failure. In *2016 IEEE 14th International Symposium on Applied Machine Intelligence and Informatics (SAMII)*, pages 165–170, Herlany, Slovakia, Jan. 2016. IEEE. ISBN 978-1-4673-8740-8. doi: 10.1109/SAMI.2016.7423001. URL <http://ieeexplore.ieee.org/document/7423001/>.

- [17] S. E. Sorour, T. Mine, K. Goda, and S. Hirokawa. A Predictive Model to Evaluate Student Performance. *Journal of Information Processing*, 23(2):192–201, 2015. ISSN 1882-6652. doi: 10.2197/ipsjjip.23.192. URL https://www.jstage.jst.go.jp/article/ipsjjip/23/2/23_192/_article.
- [18] J. Spacco, W. Pugh, N. Ayewah, and D. Hove-meyer. The Marmoset project: an automated snapshot, submission, and testing system. In *Companion to the 21st ACM SIGPLAN conference on Object-oriented programming systems, languages, and applications - OOPSLA '06*, page 669, Portland, Oregon, USA, 2006. ACM Press. ISBN 978-1-59593-491-8. doi: 10.1145/1176617.1176665. URL <http://portal.acm.org/citation.cfm?doid=1176617.1176665>.
- [19] D. Steinley. Properties of the Hubert-Arable Adjusted Rand Index. *Psychological Methods*, 9(3):386–396, 2004. ISSN 1939-1463, 1082-989X. doi: 10.1037/1082-989X.9.3.386. URL <http://doi.apa.org/getdoi.cfm?doi=10.1037/1082-989X.9.3.386>.
- [20] R. Venant, K. Sharma, P. Vidal, P. Dillenbourg, and J. Broisin. Using Sequential Pattern Mining to Explore Learners Behaviors and Evaluate Their Correlation with Performance in Inquiry-Based Learning. In . Lavoué, H. Drachsler, K. Verbert, J. Broisin, and M. Pérez-Sanagustín, editors, *Data Driven Approaches in Digital Education*, volume 10474, pages 286–299. Springer International Publishing, Cham, 2017. ISBN 978-3-319-66609-9 978-3-319-66610-5. doi: 10.1007/978-3-319-66610-5_21. URL http://link.springer.com/10.1007/978-3-319-66610-5_21. Series Title: Lecture Notes in Computer Science.

Format-Aware Item Response Theory for Predicting Vocabulary Proficiency

Boxuan Ma
OpenDNA Inc.
boxuan@open-dna.jp

Gayan Prasad Hettiarachchi
OpenDNA Inc.
gayan@open-dna.jp

Yuji Ando
OpenDNA Inc.
ando@open-dna.jp

ABSTRACT

Vocabulary proficiency testing plays a vital role in identifying the learner's level of vocabulary knowledge, which can be used to provide personalized materials and feedback in language-learning applications. Item Response Theory (IRT) is a classical method that can provide interpretable parameters, such as the learner's ability, question discrimination, and question difficulty in many language-proficiency testing environments. Many vocabulary proficiency tests include more than one type of question format. However, traditional IRT lacks the capability to tap into the information present within question texts and question formats, which can be ideally used to gauge a learner's underlying skills in more detail. In addressing this, we propose a model to reinforce traditional IRT with deep learning to exploit the information hidden within question content and format. Experimental results on a sample real-world dataset demonstrate the effectiveness of the proposed model, highlighting that question-related information can be utilized to predict a learner's performance more accurately.

Keywords

Item response theory, Deep learning, Item difficulty, English vocabulary.

1. INTRODUCTION

Vocabulary proficiency assessment plays an important part in language education and has lately gained increased popularity in online language learning. It is crucial to identify the learners' English vocabulary proficiency to higher accuracy in providing personalized materials and adaptive feedback in language-learning applications [1]. With the estimated learners' vocabulary knowledge state, systems can better gauge the attainment levels of learners and tailor the learning materials accordingly. Moreover, learners may also develop better learning plans to deal with their specific weaknesses and maximize their learning efficacy depending on the results. Most importantly, it can help place a second-language learner quickly in the ideal content space when returning to the application after a long break during which he or she may have forgotten a lot or, conversely, have progressed in the target language outside the realm of the application [2].

Computerized adaptive testing (CAT) is a mode of testing that has gained popularity because of its unparalleled ability to measure latent abilities in large-scale testing environments [3]. In CAT,

estimating the difficulty level, also called item calibration, is essential for maintaining, updating, and developing new items for an item bank. Item Response Theory (IRT) [4] is a classical method widely used to determine item difficulties. IRT can predict student performance using the logistic-like item response function and provide interpretable parameters. For this reason, different IRT models have been widely applied in CAT applications [5].

Although IRT has made a great deal of success and is widely applied, some problems still limit its usefulness. The critical drawback of traditional IRT is that it can only exploit the response results and ignore the actual contents and formats of the items [6]. Thus, IRT cannot capture the rich information hidden within question texts and underlying formats. This problem leaves no possibility of generalizing item parameters to unseen items and understanding the format's impact on the difficulty of items [2]. In addition, IRT only provides an overall latent trait for learners, while each question usually assesses different knowledge concepts or skills [7]. Thus, enhancing IRT to provide detailed results on each knowledge concept or skill in a reliable way is still an open issue.

Many researchers are beginning to focus on new approaches for estimating the difficulty of questions or items to improve traditional IRT. Studies have already shown that the representational information of questions is significantly related to the difficulty level. For English vocabulary questions, word length and corpus frequency prove to be essential factors for predicting vocabulary difficulty [8], while the average word and sentence lengths have been used as key features to predict English text difficulty [9]. Along these lines, many works have begun to estimate difficulty parameters based on items' textual content using deep neural networks [2].

In vocabulary proficiency assessment, some studies have indicated that even for the same vocabulary item, different question formats impact the difficulty level and explanatory power in predicting receptive skills [10]. The ability of learners to fully comprehend a specific word can be divided into different components. The best-known and most widely used framework is Nation's division of vocabulary knowledge into nine components of 'word knowledge' (e.g., spelling, word parts, meaning, grammatical functions, and collocation) [11]. The framework has been instrumental in describing the totality of what learners need to know. However, no single question format can adequately describe vocabulary comprehension. Usually, different question formats are used to assess different skills, such as learners' reading, writing, listening, and speaking skills collectively. However, IRT only provides an overall latent trait on the question level and cannot provide more detailed results on the underlying skills.

B. Ma, G. P. Hettiarachchi, and Y. Ando. Format-aware item response theory for predicting vocabulary proficiency. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 695–700, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853091>

2. RELATED WORK

IRT is one of the most time-tested theories for estimating latent abilities and has been used in educational testing environments since the 1950s [12]. There are several IRT models widely in use, such as the 1-parameter, 2-parameter, and 3-parameter models [13,14,15]. Extended from IRT, Multidimensional Item Response Theory (MIRT) [17] tries to meet multidimensional data demands by including an individual's multidimensional latent abilities for each skill. Although MIRT goes a step further to include the knowledge-concept proficiencies of individuals, it is sensitive to the knowledge concepts on which they have high latent abilities [17]. In addition, since the process of estimating the parameters for MIRT is the same as IRT, these two models share the same shortcomings.

With the recent surge in interest in deep learning, many works have begun to incorporate deep learning models into IRT to address the shortcomings of traditional IRT models. For example, the synthesis of Bayesian knowledge tracing (BKT) and IRT [18, 19] empowers the individualization of questions and learners. Recently, Deep-IRT [20] was proposed by combining a dynamic key-value memory network (DKVMN) [21] with an IRT module to improve the explanatory capabilities of the parameters. Furthermore, Emiko et al. [22] improved Deep-IRT with two independent neural networks for students and items.

Other IRT-based works have focused on improving the estimation accuracy of parameters by exploiting the semantic representations from question texts. Cheng and Liu [23] proposed a general Deep Item Response Theory (DIRT) framework that uses deep learning to estimate item discrimination and difficulty parameters by extracting information from item texts. Benedetto et al. [24] adopted transfer learning on Transformer language models [25] and performed the estimation of the difficulty parameter. Hsu et al. [26] proposed a method for automated estimation of multiple-choice items' difficulty for social studies tests. Their findings suggest that the semantic similarity between a stem and the options strongly impacts item difficulty. Susanti et al. [27] proposed a system for automatically generating questions for vocabulary tests. Factors such as the reading passage difficulty, semantic similarity between the correct answer and distractors, and distractor word difficulty level are all considered for controlling generated items' difficulty in this system.

Studies looking into language tests also tried to predict the item difficulty and automatically generate items of various difficulty levels. Many of these studies have investigated the relationship between test item difficulty and linguistic features such as passage length, word length, and word frequency. Hoshino and Nakagawa [28] used a support vector machine to estimate the difficulty of cloze items for a CAT. Beinborn et al. [29] used Natural Language Processing (NLP) to predict c-test difficulty at the word-gap level, using a combination of factors such as phonetic difficulty and text complexity. Loukina et al. [30] conducted a study to investigate which textual properties of a question affect the difficulty of listening items in an English language test. Settles et al. [31] used Machine Learning and NLP to induce proficiency scales and then used linguistic models to estimate item difficulty directly for CAT. However, these studies did not consider a variety of item formats that would typically appear in a test, and failed to consider linguistic skills in vocabulary learning. Recently, Brian and Andrew [48] have incorporated rich linguistic features (lexical, morphological, and syntactic features) as skills in skill-based models for learners' vocabulary learning performance prediction. It highlighted that the use of linguistic skills is quite

helpful in this regard. However, their work also failed to consider question formats' influence on the difficulty level and different receptive skills. In addressing this, here we incorporate item-format information and associated skill requirements to improve the estimations of IRT parameters, and in effect, the prediction accuracy of a learners' performance.

3. PROPOSED METHOD

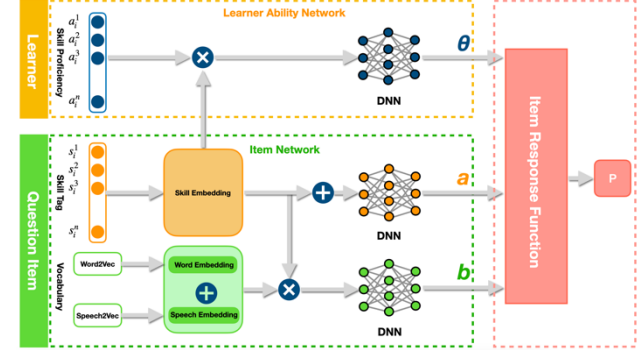


Figure 1. Overview of the proposed framework.

3.1 Framework

Inspired by previous studies [6, 23], we propose a framework to enhance traditional IRT with deep learning, which aims to obtain the learner parameter (ability) and item parameters (discrimination and difficulty) to predict learner performance in vocabulary questions. In achieving this, as shown in Figure 1, our framework comes with three parts: the learner ability network, item network, and prediction module.

3.1.1 Item Network

The items' characteristics, i.e., the difficulty and discrimination parameters, are calculated in the item network.

For a vocabulary question presented in a specific format, two elements influence the item's characteristics: the target vocabulary and the required skills to respond correctly. For the target vocabulary, the semantic features are embedded into a d -dimensional vector v using pre-trained Word2Vec [32] vector v_w and Speech2Vec [33] vector v_s , where $v = v_w \oplus v_s$ and $d = 50$. And the required skills for this question are represented by one-hot vectors $S = (S_1, S_2, \dots, S_n)$, $S_i \in \{0,1\}^n$, where n is the number of required skills. Then, we utilize a d -dimensional dense layer to acquire the dense embedding for each skill S_i for training, the dense embedding of S_i as s_i , and $s_i \in \mathbb{R}^d$:

$$s_i = S_i W_s, \quad (1)$$

where $W_s \in \mathbb{R}^{p \times d}$ are the parameters of the dense layer. Then we use the target vocabulary embedding and the skill embedding to obtain the item parameters (discrimination and difficulty).

Discrimination. Question discrimination a can be used to analyze learner performance distribution on the question. Inspired by previous works [17, 23], we learn a from required skills that correspond to the question. A deep neural network (DNN) is trained to estimate a . Specifically, we sum the dense embedding of required skills to get a d -dimensional vector $A \in \mathbb{R}^d$. Then, we input A into the DNN to estimate a . Finally, we normalize a so that it is in the range $[-4, 4]$ [16]. The definition of a is as follows:

$$a = 8 \times (\text{sigmoid}(\text{DNN}_a(A)) - 0.5), A = s \oplus s. \quad (2)$$

Difficulty. Question difficulty b determines how hard the question is. Adopting from previous works [8, 9], we predict b based on the semantic features of the target word. In addition, the depth and width of the required skills examined by the question also significantly impact the difficulty. The deeper and broader the required skills being examined, the more difficult the question is [23]. Therefore, we adopt a DNN to model b based on the target vocabulary embedding v and the required skills s depending on the corresponding question format. Like the discrimination, we normalize b so that it is in the range $[-4, 4]$ [16]. The definition of b is as follows:

$$b = 8 \times (\text{sigmoid}(DNN_b(B)) - 0.5), \quad B = s \odot v. \quad (3)$$

3.1.2 Learner Ability Network

In the learner network, the proposed method calculates a learner’s ability. For a learner, we initialize a proficiency vector $\alpha = (\alpha_1, \alpha_2, \dots, \alpha_n)$ randomly, where $\alpha_i \in [0, 1]$ represents the degree of proficiency of a learner on a specific skill i .

Ability. Learners’ ability θ has strong interpretability for their performance on questions. It is closely related to the proficiency of various skills tested in the questions [23]. Therefore, we multiply the corresponding proficiency α with the skill dense embedding of the questions s and get a d -dimensional vector $\theta \in \mathbb{R}^d$. Then we input θ into a DNN to learn the ability parameter, which is defined as follows:

$$\theta = DNN\theta(\theta), \quad \theta = \alpha \odot s. \quad (4)$$

3.1.3 Prediction of Learner Response

Like previous works [20, 22, 23], the proposed method predicts a learner’s response performance to a question as a probability. We input the trained parameters, namely, θ , a , and b into the item response function (Eq.5) to predict $P(\theta)$, the learner’s probability of answering the specific question correctly.

$$P(\theta) = \frac{1}{1 + e^{-a(\theta - b)}}. \quad (5)$$

3.1.4 Model Learning

The parameters to be updated in the proposed framework mainly exist in two parts: learner ability network and item network. The updating parameters include the proficiency vector α and skill dense embedding weights W_s . In addition, the weights of the three DNNs $\{W_{DNN_a}, W_{DNN_b}, W_{DNN_\theta}\}$ are updated as well.

The loss function of the proposed method is the negative log-likelihood function. The learner’s response is recorded as 1 when he/she answers the item correctly and 0 otherwise. For learner i and question j , let r_{ij} be the actual score for learner i on question j , and \tilde{r}_{ij} be the predicted score. Thus, the loss for learner i on question j is defined as:

$$\mathcal{L} = r_{ij} \log \tilde{r}_{ij} + (1 - r_{ij}) \log(1 - \tilde{r}_{ij}). \quad (6)$$

Using Adam optimization [34], all parameters are learned simultaneously by directly minimizing the objective function.

4. EVALUATION

4.1 Dataset

Our real-world dataset came from one of Japan’s most popular English-language learning applications. We tentatively used a sample dataset from 129 application users who newly registered in 2021, and most of them are Japanese students learning English. This dataset included 1,900 English words labeled by the

Common European Framework of Reference for Languages (CEFR), mainly in the B1/B2 range. Each word in the dataset had six different question types collectively assessing reading, writing, listening, and speaking skills. The dataset included the initial responses (when encountering for the first time) of the users to such questions.

4.1.1 Item Formats

The knowledge pertaining to English words is not all-or-none as with the case with any other language. Rather, there are different aspects, such as knowledge of the reading, writing, listening, speaking, grammatical behavior, collocation behavior, word frequency, stylistic register constraints, conceptual meaning, the associations a word has with other related words, and so on [11, 35]. Hence, as summarized in Table 1, there are six different question formats to collectively assess reading, writing, listening, and speaking skills of vocabulary learning in our dataset. For each format, we indicate the linguistic skill(s) required to tackle the question (L = listening, R = reading, S = speaking, W = writing) and some of the evidence from the literature supporting this assignment. Below are the descriptions of the six question formats. Multiple-choice definition: choose the Japanese description of the English word. Multiple-choice recall: choose the corresponding English word given the Japanese description. Spelling: type the spelling of the English word given the Japanese description. Cloze test with spelling: type in the blank with the appropriate English word. Multiple-choice listening: choose the corresponding English word given the pronunciation. Multiple-choice cloze test: choose the appropriate English word to fill the blank.

Table 1. Summary of question formats and required skill(s).

Label	Question Format	Skills	References
Format 1	Multiple-choice definition	R	[40,41,44]
Format 2	Multiple-choice recall	R	[40,41]
Format 3	Spelling	R S W	[39]
Format 4	Cloze test with spelling	R S W	[39,42]
Format 5	Multiple-choice listening	L R	[38,43]
Format 6	Multiple-choice cloze test	R W	[31,39,42]

4.2 Experimental Settings

We conducted extensive experiments to evaluate the accuracy of our model in predicting the performance of learners and compared it with several existing models. To set up the experiments, we partitioned the dataset, where the question-user interactions were divided into the training and testing sets at different ratios: 60%, 70%, 80%, and 90%.

We name our method Format-Aware IRT(FIRT). FIRT-S is a variant of FIRT, which only uses speech embedding based on the Speech2Vec, and FIRT-W is a variant that only uses word embedding based on the Word2Vec. We compared our method’s performance with IRT, MIRT, and Probabilistic matrix factorization (PMF) [36].

Following previous works [23, 37], we chose four widely used metrics for the evaluation: Prediction Accuracy (ACC), Area Under Curve (AUC), Mean Absolute Error (MAE), and Root Mean Square Error (RMSE). The smaller the values of RMSE and MAE, and the larger the values of AUC and ACC, the better the results.

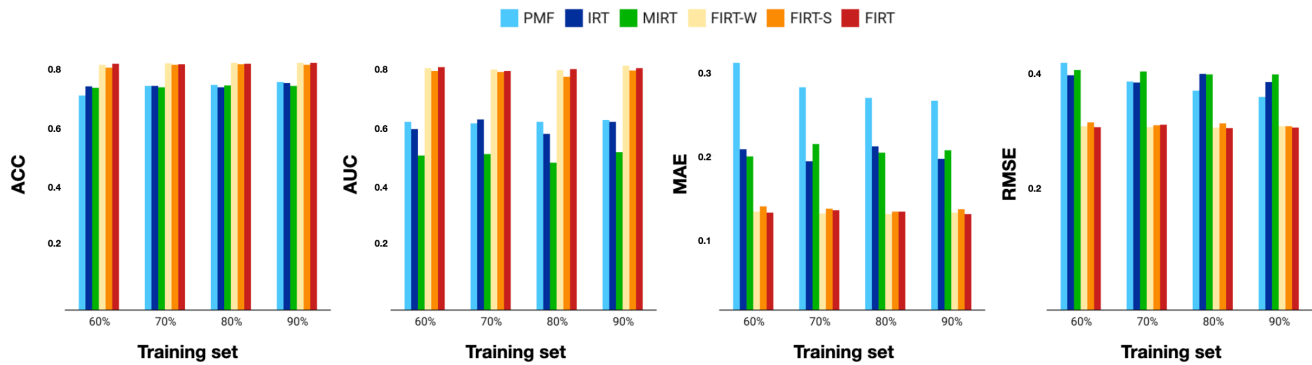


Figure 2. Comparison of learner performance prediction among different methods.

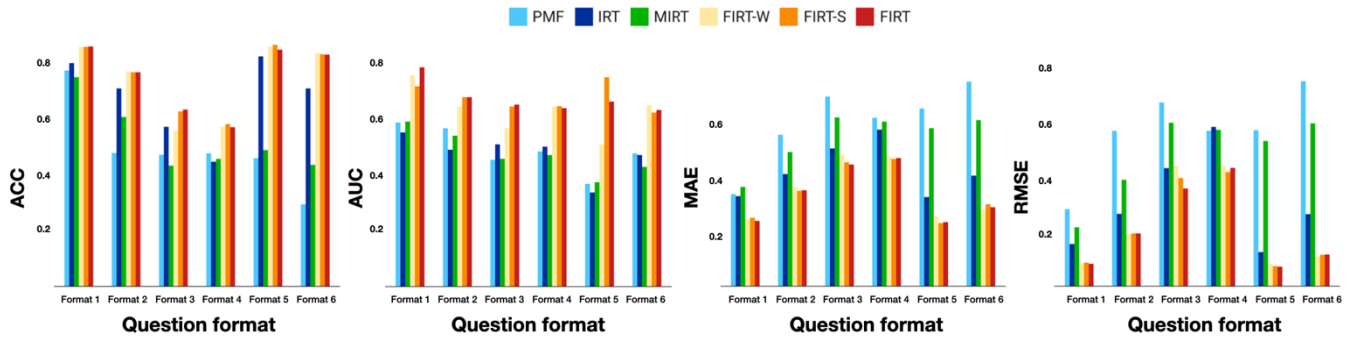


Figure 3. Comparison among different question formats.

4.3 Results

4.3.1 Performance Prediction

The overall results on all four metrics are shown in Figure 2 for six different models predicting learners' performance. We observe that our proposed models (FIRT, FIRT-S, and FIRT-W) perform better than other traditional baseline models, such as IRT, MIRT, and PMF. It is clear that our deep learning-based models can effectively make use of the vocabulary content and format information to improve the performance.

4.3.2 Impact of Different Formats

Many studies predicting vocabulary knowledge use only a single question format. However, results based on a single format can be misleading because it might be gauging a limited skill space pertaining to vocabulary knowledge. Understanding the differences among various formats may provide insight for developing tests and tools that can measure proficiency on a much-balanced scale [47]. Along this line, we conducted experiments to evaluate our model's ability to predict performance on different question formats, and to illustrate the variability of performance depending on the format.

The results in Figure 3 show that our models perform better in all question formats. This indicates that the vocabulary content and format information, together with the underlying skill proficiency, help predict learners' performance better, which are typically ignored in the traditional methods. Also, we observe that the prediction performance is strongly affected by the question format. As we mentioned earlier, different question formats assess different linguistic skills like reading, writing, listening, and speaking. The results show that all models have considerably satisfactory performance for multiple-choice items, which assess only one or two underlying skills and are easy to understand and

answer [45, 46]. However, the prediction performance for Format 3 (Spelling) and Format 4 (Cloze test with spelling) are deficient compared with others, which intuitively suggests that responses to question formats that necessitate multiple underlying skills are more difficult to predict accurately. In addition, we noticed that FIRT-S (using speech embedding) performs slightly better than other models for Format 5 (Multiple-choice listening). This implies that using other features besides semantic features may improve the performance. Moreover, the findings implies that testers should consider the effect of different formats when assessing vocabulary knowledge and strive to use a combination of formats in vocabulary assessment to gauge a broader skill space.

5. CONCLUSION

In this work, we proposed a framework that reinforces IRT with deep learning routines that take full advantage of the questions' representational information, such as the question contents, formats, and the required linguistic skill(s) to tackle the question. Experiments were conducted to confirm the effectiveness of the proposed approach, and the results showed that our method performs better than other methods. We highlight that vocabulary content and format information together with the required skill set is useful in accurately predicting learners' vocabulary proficiency.

However, there are some limitations in this work. The dataset is relatively small, and the learner base is limited to learners of the same language background. For future work, we plan to collect more data on learners of various backgrounds, which may be useful when generalizing the method to a broader audience. Also, it is likely that the six item formats explored in this work over-index on language reception skills rather than production skills (i.e., writing and speaking). In going forward, we need to test more writing and speaking questions, and include additional linguistic skills to expand the capabilities of our model.

6. REFERENCES

- [1] Avdiu, D. and Bui, V. 2019. Predicting learner knowledge of individual words using machine learning. In *Proceedings of the 8th Workshop on NLP for Computer Assisted Language Learning*, pages 1-9.
- [2] Robertson, F. 2021. Word Discriminations for Vocabulary Inventory Prediction. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, pages 1188-1195.
- [3] Wainer, H., Dorans, N. J., Flaughner, R., Green, B. F., & Mislevy, R. J. 2000. *Computerized adaptive testing: A primer*. Routledge.
- [4] Embretson, Susan E., and Steven P. 2013. *Reise. Item response theory*. Psychology Press.
- [5] Hambleton, R. K. 1989. Principles and selected applications of item response theory.
- [6] Liu, Q., Huang, Z., Yin, Y., Chen, E., Xiong, H., Su, Y., and Hu, G. 2019. Ekt: Exercise-aware knowledge tracing for student performance prediction. *IEEE Transactions on Knowledge and Data Engineering*, 33(1), 100-115.
- [7] Huang, Z., Liu, Q., Chen, E., Zhao, H., Gao, M., Wei, S., ... and Hu, G. 2017. Question Difficulty Prediction for READING Problems in Standard Tests. In *Thirty-First AAAI Conference on Artificial Intelligence*, pages 1352-1359.
- [8] Culligan, B. 2015. A comparison of three test formats to assess word difficulty. *Language Testing*, 32(4), 503-520.
- [9] Beinborn, L., Zesch, T., and Gurevych, I. 2014. Predicting the difficulty of language proficiency tests. *Transactions of the Association for Computational Linguistics*, 2, 517-530.
- [10] Kremmel, B., and Schmitt, N. 2016. Interpreting vocabulary test scores: What do various item formats tell us about learners' ability to employ words?. *Language Assessment Quarterly*, 13(4), 377-392.
- [11] Nation, I. S. 2001. *Learning vocabulary in another language*. Cambridge university press.
- [12] Embretson, S. E., and Reise, S. P. 2013. *Item response theory*. Psychology Press.
- [13] Baker, F. B., & Kim, S. H. 2004. *Item Response Theory: Parameter Estimation Techniques*, Second Edition. *Statistics: A Series of Textbooks and Monographs*. Taylor & Francis.
- [14] Lord, F. M., and Novick, M. R. 1968. *Statistical Theories of Mental Test Scores*. Addison-Wesley.
- [15] Ueno, M., and Miyasawa, Y. 2015. Probability based scaffolding system with fading. *Artificial Intelligence in Education. 17th International Conference, AIED*, pages 237-246.
- [16] Van der Linden, W. J., and Hambleton, R. K. (1997). *Handbook of item response theory*. Taylor & Francis Group. Cited on page 1(7), 8.
- [17] Yao, L., & Schwarz, R. D. 2006. A multidimensional partial credit model with associated item and test statistics: An application to mixed-format tests. *Applied psychological measurement*, 30(6), 469-492.
- [18] Mohammad M. Khajah, Yun Huang, Jos'e P. Gonz'alez-Brenes, Michael C. Mozer, and Peter Brusilovsky. 2014. Integrating knowledge tracing and item response theory: A tale of two frameworks. In *Proceedings of the 4th International Workshop on Personalization Approaches in Learning Environment*, pages 7-15.
- [19] Kevin H. Wilson, Yan Karklin, Bojian Han, and Chaitanya Ekanadham. 2016. Back to the basics: Bayesian extensions of IRT outperform neural networks for proficiency estimation. In *Proceedings of the 9th International Conference on Educational Data Mining*, pages 539-544.
- [20] C. Yeung. 2019. Deep-irt: Make deep learning based knowledge tracing explainable using item response theory. In *Proceedings of the 12th International Conference on Educational Data Mining*, pages 683-686.
- [21] Zhang, J., Shi, X., King, I., and Yeung, D. Y. 2017. Dynamic key-value memory networks for knowledge tracing. In *Proceedings of the 26th international conference on World Wide Web*. pages 765-774.
- [22] Tsutsumi, E., Kinoshita, R., and Ueno, M. 2021. Deep-IRT with Independent Student and Item Networks. In *Proceedings of the 14th International Conference on Educational Data Mining*, pages 510-517.
- [23] Cheng, S., Liu, Q., Chen, E., Huang, Z., Huang, Z., Chen, Y., and Hu, G. 2019. DIRT: Deep learning enhanced item response theory for cognitive diagnosis. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. pages 2397-2400.
- [24] Benedetto, L., Aradelli, G., Cremonesi, P., Cappelli, A., Giussani, A., and Turrin, R. 2021. On the application of Transformers for estimating the difficulty of Multiple-Choice Questions from text. In *Proceedings of the 16th Workshop on Innovative Use of NLP for Building Educational Applications*. pages 147-157.
- [25] Dai, Z., Yang, Z., Yang, Y., Carbonell, J. G., Le, Q., and Salakhutdinov, R. 2019. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. pages 2978-2988.
- [26] Hsu, F. Y., Lee, H. M., Chang, T. H., and Sung, Y. T. 2018. Automated estimation of item difficulty for multiple-choice tests: An application of word embedding techniques. *Information Processing & Management*, 54(6), 969-984.
- [27] Susanti, Y., Nishikawa, H., Tokunaga, T., and Hiroyuki, O. 2016. Item Difficulty Analysis of English Vocabulary Questions. In *International Conference on Computer Supported Education*. Vol. 2, pages. 267-274.
- [28] Hoshino, A., and Nakagawa, H. 2010. Predicting the difficulty of multiple-choice close questions for computer-adaptive testing. *Natural Language Processing and its Applications*, 46, pages 279-292.
- [29] Beinborn, L., Zesch, T., and Gurevych, I. 2014. Predicting the difficulty of language proficiency tests. *Transactions of the Association for Computational Linguistics*, 2, 517-530.
- [30] Loukina, A., Yoon, S. Y., Sakano, J., Wei, Y., and Sheehan, K. 2016. Textual complexity as a predictor of difficulty of listening items in language proficiency tests. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. pages 3245-3253.

- [31] Settles, B., T LaFlair, G., and Hagiwara, M. 2020. Machine learning–driven language assessment. *Transactions of the Association for computational Linguistics*, 8, 247-263.
- [32] Mikolov, T., Chen, K., Corrado, G., and Dean, J. 2013. Efficient Estimation of Word Representations in Vector Space. *arXiv preprint arXiv:1301.3781*.
- [33] Chung, Y. A., and Glass, J. 2018. Speech2vec: A sequence-to-sequence framework for learning word embeddings from speech. *arXiv preprint arXiv:1803.08976*.
- [34] Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- [35] Dale, E. 1965. Vocabulary measurement: Techniques and major findings. *Elementary english*, 42(8), 895-948.
- [36] Mnih, A., and Salakhutdinov, R. R. 2007. Probabilistic matrix factorization. *Advances in neural information processing systems*, 20.
- [37] Wu, R., Xu, G., Chen, E., Liu, Q., and Ng, W. 2017. Knowledge or gaming? Cognitive modelling based on multiple-attempt response. In *Proceedings of the 26th International Conference on World Wide Web Companion*. pages 321-329.
- [38] Bradlow, A. R., and Bent, T. 2002. The clear speech effect for non-native listeners. *The Journal of the Acoustical Society of America*, 112(1), 272-284.
- [39] Khodadady, E. 2014. Construct validity of C-Tests: A factorial approach. *Journal of Language Teaching and Research*, 5(6), 1353.
- [40] Staehr, L. S. 2008. Vocabulary size and the skills of listening, reading and writing. *Language Learning Journal*, 36(2), 139-152.
- [41] Milton, J. 2010. The development of vocabulary breadth across the CEFR levels. *Communicative proficiency and linguistic development: Intersections between SLA and language testing research*, 211-232.
- [42] Klein-Braley, C. 1997. C-Tests in the context of reduced redundancy testing: An appraisal. *Language testing*, 14(1), pages 47-84.
- [43] Milton, J., Wade, J., and Hopkins, N. 2010. Aural word recognition and oral competence in English as a foreign language. *Insights into non-native vocabulary teaching and learning*, 83-98.
- [44] Brown, J., Frishkoff, G., and Eskenazi, M. 2005. Automatic question generation for vocabulary assessment. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*. pages 819-826.
- [45] Kremmel, B., and Schmitt, N. 2016. Interpreting vocabulary test scores: What do various item formats tell us about learners' ability to employ words?. *Language Assessment Quarterly*, 13(4), 377-392.
- [46] Kilickaya, F. (2019). Assessing L2 vocabulary through multiple-choice, matching, gap-fill, and word formation items. *Lublin Studies in Modern Languages and Literature*, 43(3), 155-166.
- [47] Bowles, R. P., and Salthouse, T. A. 2008. Vocabulary test format and differential relations to age. *Psychology and Aging*, 23(2), 366.
- [48] Zylich, B., and Lan, A. 2021. Linguistic Skill Modeling for Second Language Acquisition. In *LAK21: 11th International Learning Analytics and Knowledge Conference*. pages 141-150.

Towards Automated Generation and Evaluation of Questions in Educational Domains

Shravya Bhat
Carnegie Mellon University
shravyab@andrew.cmu.edu

Huy A. Nguyen
Carnegie Mellon University
hn1@cs.cmu.edu

Steven Moore
Carnegie Mellon University
stevenmo@andrew.cmu.edu

John Stamper
Carnegie Mellon University
jstamper@cs.cmu.edu

Majd Sakr
Carnegie Mellon University
msakr@cs.cmu.edu

Eric Nyberg
Carnegie Mellon University
ehn@cs.cmu.edu

ABSTRACT

Students learn more from doing activities and practicing their skills on assessments, yet it can be challenging and time consuming to generate such practice opportunities. In this work, we present a pipeline for generating and evaluating questions from text-based learning materials in an introductory data science course. The pipeline includes applying a T5 question generation model and a concept hierarchy extraction model on the text content, then scoring the generated questions based on their relevance to the extracted key concepts. We further classified the generated questions as either useful to learning or not with two different approaches: automated labeling by a trained GPT-3 model and manual review by expert human judges. Our results showed that the generated questions were rated favorably by all three evaluation methods. We conclude with a discussion of the strengths and weaknesses of the generated questions and outline the next steps towards refining the pipeline and promoting NLP research in educational domains.

1. INTRODUCTION

As education across grade levels continues to transition towards online platforms in response to the COVID-19 pandemic, the need for effective and scalable assessment tools emerges as a pressing issue for instructors and educators. Amid many other logistical issues that arise from emergency online education [5], instructors often find themselves having to generate a large question bank to accommodate this new learning format. In turn, this challenge motivates the need for supporting instructor efforts via methods that automatically generate usable assessment questions based on the learning materials, in a way that requires minimal inputs from instructors and domain experts.

S. Bhat, H. Nguyen, S. Moore, J. Stamper, M. Sakr, and E. Nyberg. Towards automated generation and evaluation of questions in educational domains. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 701–704, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853085>

Recent advances in natural language processing (NLP), question answering and question generation (QG) offer a promising path to accomplishing this goal. Most theories of learning emphasize repeated practice as an important mechanism for mastering low-level knowledge components, which altogether contribute to the high-level learning objectives [7]. We therefore envision that having the ability to generate questions on-demand would accommodate students' varying levels of learning needs, while allowing instructors to allocate resources to other components of the course. Our work presents an initial step towards realizing this capability. We applied Text-To-Text Transfer Transformer (T5) models [15] on conceptual reading materials from a graduate-level data science course to generate potential questions that may be used for assessment. We then evaluated these questions in three different ways. First, we conducted a separate concept hierarchy extraction process on the reading materials to extract the important concept keywords and scored each generated question based on how many such keywords it contains. Second, we applied a fine-tuned GPT-3 model to classify the questions as either useful to learning or not. Finally, we had two data science instructors perform this same classification task manually. Our results contribute insights into the feasibility of applying state-of-the-art NLP models in generating meaningful questions, with a pipeline that generalizes well across learning domains.

2. METHODS

2.1 Dataset

We used the learning materials from a graduate-level introductory data science course at an R1 university in the northeastern United States. The course has been offered every semester since Summer 2020, with class sizes ranging from 30-90 in general. The course content is divided into the conceptual components and the hands-on projects. Students learn from six conceptual units, further broken down into sixteen modules, each corresponding to a data science topic such as *Feature Engineering* and *Bias-Variance Trade-off*. Each module consists of reading assignments, ungraded formative assessments and weekly quizzes serving as graded summative assessments. Students also get to practice with the learned concepts through seven hands-on coding projects, which are evaluated by an automatic grading system. In the scope of this work, we will focus on generating

questions from the textual content of the sixteen modules in the course, using the following pipeline.

2.2 Question Generation Pipeline

First, we extracted the learning materials from an online learning platform which hosts the course. This extracted data is in XML format, which preserves not only the text content but also its hierarchy within the course structure (i.e., which module and unit each paragraph belongs to). We scraped the text content from the XML files using the BeautifulSoup¹ library and cleaned the content to remove leading questions, such as “*What does this accomplish*” and “*Why would this make sense?*”. These questions were included to help students navigate the reading more effectively but do not contain meaningful information on their own. From this point, the resulting text data was input to two separate processes as follows.

Concept Hierarchy Extraction. This process was carried out by the MOOCCubeX pipeline [16], which performs weakly supervised fine-grained concept extraction on a given corpus without relying on expert input. As an example, given a paragraph that explains Regression, some of the extracted concepts include *least-squared error*, *regularization*, and *conditional expectation*; these could be viewed as the key concepts which students are expected to understand after reading the materials. A researcher in the team reviewed the generated concepts and manually removed those which were deemed invalid, including prepositions (e.g., ‘around’), generic verbs (e.g., ‘classifying’) and numbers (e.g., ‘45’ – this is part of a numeric example in the text, rather than an important constant to memorize).

Question Generation. For this process, we applied Google’s T5 [15], a transformer-based encoder-decoder model. Since its pre-training involves a multi-task structure of supervised and unsupervised learning, T5 works well on a variety of natural language tasks by merely changing the structure of the input passed to it. For our use case, the input data is the cleaned text content prepended by a header of the text. Our rationale for including the header is to inform the model of the high level concept which the generated questions should center around. We had previously tried extracting answers from the text content using a custom rule-based approach with a dependency parse tree, but found that this resulted in the creation of more nonsensical than sensible questions; in comparison, incorporating the headers led to higher quality questions. There were three hierarchical levels of header that were used in our input: Unit, Module and Title, where the former encompasses the latter. For example, the unit *Exploratory Data Analysis* includes the module *Feature Engineering*, which has a section titled *Principal Component Analysis*, among others. Before applying the model to our dataset, we also fine-tuned it on SQuAD 1.1, a well known reading comprehension dataset and a common benchmark for question-answering models [11].

2.3 Evaluation

We evaluated the generated questions with three different methods as follows.

¹<https://www.crummy.com/software/BeautifulSoup/bs4/doc/>

Information Score. This is a custom metric that denotes how relevant each question is to the key concepts identified in the Concept Hierarchy Extraction step. We denote this set of key concepts as C . For every generated question q , we further denote $T(q)$ as the set of tokens in it and compute the *information score* as the number of tokens in q that coincide with an extracted concept,

$$IS(q) = \frac{1}{|T(q)|} \sum_{t \in T(q)} 1(t \in C), \quad (1)$$

where the division by $|T(q)|$ is for normalization. With this formulation, higher scores indicate better questions that touch on more of the key learning concepts.

GPT-3 Classification. We used a GPT-3 model as it has been a popular choice for text classification tasks such as detecting hate speech [3] and text sentiment [17]. Our classification task involves rating each generated question as either *useful for learning* or *not useful*. A useful-for-learning question is one that pertains to the course content and is intended to assess the domain knowledge of the student. On the other hand, a question is classified as not useful if it is vague, unclear, or not about assessing domain knowledge. For example, the question “*What programming language do I need to learn before I start learning algorithms?*” is a valid question, but it is classified as not useful for learning because it pertains to a course prerequisite rather than domain knowledge assessment. To perform this classification, we first fine-tuned the GPT-3 model with default hyperparameters on the LearningQ dataset [2], which contains 5600 student-generated questions from Khan Academy. Each question contains a label to indicate if it is useful for learning or not, as annotated by two expert instructors. Next, we passed in the T5-generated questions as the GPT-3 model’s input, obtaining the output as a set of binary labels indicating if it rated each question as useful for learning or not.

Expert Evaluation. To further validate the question quality, we had two expert raters with 5+ years of teaching experience in the domain of data science rate each question. Following the same classification process as in [2], the two raters indicated if each question was useful for learning or not. We measured the Inter-Rater Reliability (IRR) between the two raters and found they achieved a Cohen’s kappa of $\kappa = 0.425$, with similarity in 75.59% of the question ratings, indicating a moderate level of agreement [9]. The remaining discordant questions were discussed between the two raters until they reached a consensus on their classification, resulting in all of the generated questions being classified by both human judges and the GPT-3 model.

3. RESULTS

Following the above pipeline, we generated a total of 203 questions across the three header levels - Module, Unit, and Title. The Appendix shows a number of example generated questions, along with their information scores and GPT-3 model evaluation. Among the 203 questions, 151 (74.38%) were classified as useful for learning by the GPT-3 model. To compare this classification with the human raters’ consensus, we constructed a confusion matrix as shown in Table 1. We observed that the model agreed with human raters in 135 (66.50%) instances; in cases where they disagreed, most of

the mismatches (52 out of 68) were due to the GPT-3 model overestimating the questions’ usefulness.

	Expert: 0	Expert: 1
GPT-3: 0	36	16
GPT-3: 1	52	99

Table 1: Confusion matrix for comparing GPT-3 and expert ratings on the generated questions. 0 denotes Not Useful and 1 denotes Useful rating.

We followed up with a qualitative review of the questions rated as not useful by human experts to better understand (1) what separated them from the questions rated as useful, and (2) why the GPT-3 model might still rate them as useful. For (1), we identified two important requirements that a question generally needs to meet to be rated as useful by human experts. First, it has to thoroughly set up the context (e.g., what is the scenario, how many responses are expected) from which an answer could be reasonably derived. An example question that satisfies this category is “*What are **two** types of visions that a data science team will **work with a client** to develop?*,” where the bolded terms are important contextual factors which make the question useful. We further note that useful questions with thorough contexts tend to be longer, because they necessarily include more information to describe such contexts. At the same time, short questions may still be considered useful by expert raters if they target a sufficiently specific concept. For example, “*what is a way to improve a **decision tree’s performance**?*” is considered useful because the bolded term is very specific. On the other hand, a similar-looking question such as “*what is a way to analyze business data*” is not useful, due to “*analyze business data*” being too broad. The GPT-3 model typically fails to recognize this specificity criterion – many of the questions rated as useful by GPT-3, but not by human raters, are similar to ones such as “*What are two types of data science tasks?*,” which could be useful if “*data science tasks*” was replaced with a more targeted concept.

Next, we examined whether our score metric, which calculates the normalized number of important concepts that a question encapsulates, aligns with the expert classification of question usefulness for learning. We observed from Figure 1 that, across the three header levels, questions rated as useful tended to have similar or higher information scores than their counterparts.

4. DISCUSSION AND CONCLUSION

In this work, we propose and evaluate a domain-independent pipeline for generating assessment questions based on reading materials in a data science course. Our results showed that the GPT-3 model, fine tuned on the LearningQ dataset [2], was able to reach an acceptable level of agreement (on 66.50% of the questions) with the consensus of two expert raters. The model appeared to learn that long questions are likely useful, which is a reasonable assumption as these questions might contain more relevant contextual information. However, it also classified some short questions as useful, despite the lack of specificity which human evaluators could easily recognize. As the LearningQ dataset did not contain data science questions, it is no surprise that our model was not particularly good at differentiating between specific

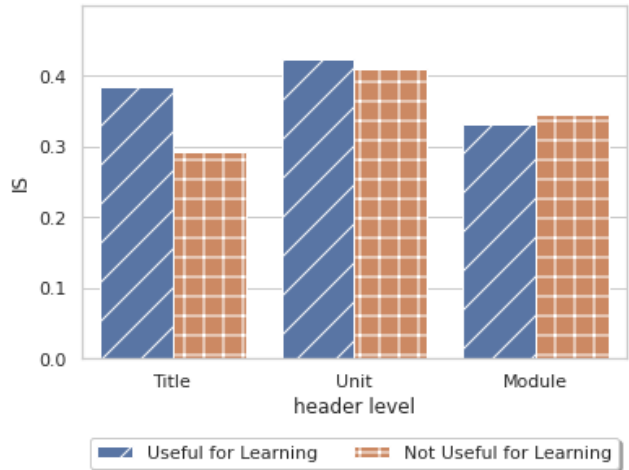


Figure 1: Distribution of information score, partitioned by expert raters’ evaluation, at each header level.

data science concepts (e.g., “*decision tree’s performance*”) and ambiguous ones (e.g., “*business data*”). Additional fine-tuning of the GPT-3 model on a labeled dataset closer to our learning domain would therefore be a promising next step.

When treating the expert rating of question usefulness as the ground truth, we found that the useful questions generally had higher information scores than those not rated as useful, suggesting that our rationale for the formulation of these metrics (i.e., that higher scores reflect more concepts captured and therefore higher quality) was justified. At the same time, several questions had relatively low information scores but were still rated as useful by experts (e.g., “*What are two types of decision trees?*”) because they target a sufficiently *specific* concept. To detect these questions, it would be beneficial to incorporate measures of the generated questions’ level of specificity [6] into the existing information score metric.

The above results have been obtained without the need for any human-labeled domain encoding, which makes our question generation pipeline highly domain-agnostic and generalizable. At the same time, there are ample opportunities to further promote its adoption across different learning domains. First, more research is needed to investigate question generation when the learning contents are not entirely textual, but may include multimedia components. Recent advances in the area of document intelligence [1, 4], combining NLP techniques with computer vision, could be helpful in this direction. Second, there remains the need to diversify the generated questions to meet a wider range of assessment goals. In particular, most of our current questions start with “what” (e.g., those in Table ??), which are primarily geared towards recalling information. Incorporating other question types in the generation pipeline could elicit more cognitive processes in Bloom’s taxonomy [8] – for example, “how” questions can promote understanding and “why” questions are designed for analyzing – which in turn contribute to better learning overall. This diversifying direction is also an area of active research in the NLP and

QG community [13, 14].

We further note that the proposed pipeline is also customizable to individual domains, so as to enable higher quality questions. First, hyperparameter tuning on a dataset relevant to the learning domain would likely improve the performance of the T5 and GPT-3 models. Second, the concept extraction process could be enhanced with a combination of machine-generated and human-evaluated skill mappings, which have been shown to result in more accurate knowledge models [10, 12]. Finally, the question evaluation criteria may also benefit from subject matter experts' inputs to closely reflect the distinct nature of the learning domain; for example, chemistry assessments could potentially include both conceptual questions (e.g., "what is the chemical formula of phenol?") and scenario-based questions (e.g., "describe the phenomenon that results from mixing sodium metal and chlorine gas?").

5. REFERENCES

- [1] D. Baviskar, S. Ahirrao, V. Potdar, and K. Kotecha. Efficient automated processing of the unstructured documents using artificial intelligence: A systematic literature review and future directions. *IEEE Access*, 2021.
- [2] G. Chen, J. Yang, C. Hauff, and G.-J. Houben. Learningq: a large-scale dataset for educational question generation. In *Twelfth International AAAI Conference on Web and Social Media*, 2018.
- [3] K.-L. Chiu and R. Alexander. Detecting hate speech with gpt-3. *arXiv preprint arXiv:2103.12407*, 2021.
- [4] B. Han, D. Burdick, D. Lewis, Y. Lu, H. Motahari, and S. Tata. Di-2021: The second document intelligence workshop. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 4127–4128, 2021.
- [5] C. B. Hodges, S. Moore, B. B. Lockee, T. Trust, and M. A. Bond. The difference between emergency remote teaching and online learning. 2020.
- [6] H. Huang, T. Kajiwara, and Y. Arase. Definition modelling for appropriate specificity. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2499–2509, 2021.
- [7] K. R. Koedinger, A. T. Corbett, and C. Perfetti. The knowledge-learning-instruction framework: Bridging the science-practice chasm to enhance robust student learning. *Cognitive science*, 36(5):757–798, 2012.
- [8] D. R. Krathwohl. A revision of bloom's taxonomy: An overview. *Theory into practice*, 41(4):212–218, 2002.
- [9] J. R. Landis and G. G. Koch. The measurement of observer agreement for categorical data. *biometrics*, pages 159–174, 1977.
- [10] R. Liu and K. R. Koedinger. Closing the loop: Automated data-driven cognitive model discoveries lead to improved instruction and learning gains. *Journal of Educational Data Mining*, 9(1):25–41, 2017.
- [11] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.
- [12] J. C. Stamper and K. R. Koedinger. Human-machine student model discovery and improvement using datashop. In *International Conference on Artificial Intelligence in Education*, pages 353–360. Springer, 2011.
- [13] M. A. Sultan, S. Chandel, R. F. Astudillo, and V. Castelli. On the importance of diversity in question generation for qa. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5651–5656, 2020.
- [14] S. Wang, Z. Wei, Z. Fan, Z. Huang, W. Sun, Q. Zhang, and X.-J. Huang. Pathq: Neural question generation from facts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9066–9075, 2020.
- [15] L. Xue, N. Constant, A. Roberts, M. Kale, R. Al-Rfou, A. Siddhant, A. Barua, and C. Raffel. mt5: A massively multilingual pre-trained text-to-text transformer. *arXiv preprint arXiv:2010.11934*, 2020.
- [16] J. Yu, Y. Wang, Q. Zhong, G. Luo, Y. Mao, K. Sun, W. Feng, W. Xu, S. Cao, K. Zeng, et al. Moocubex: A large knowledge-centered repository for adaptive learning in moocs. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 4643–4652, 2021.
- [17] R. Zhong, K. Lee, Z. Zhang, and D. Klein. Adapting language models for zero-shot learning by meta-tuning on dataset and prompt collections. *arXiv preprint arXiv:2104.04670*, 2021.

MOOC-Rec: Instructional Video Clip Recommendation for MOOC Forum Questions*

Peide Zhu
Delft University of Technology
p.zhu-1@tudelft.nl

Jie Yang
Delft University of Technology
j.yang-3@tudelft.nl

Claudia Hauff
Delft University of Technology
c.hauff@tudelft.nl

ABSTRACT

In this work, we address the information overload issue that learners in Massive Open Online Courses (MOOCs) face when attempting to close their knowledge gaps via the use of MOOC discussion forums. To this end, we investigate the recommendation of one-minute-resolution video clips given the textual similarity between the clips' transcripts and MOOC discussion forum entries. We first create a large-scale dataset from Khan Academy video transcripts and their forum discussions. We then investigate the effectiveness of applying pre-trained transformers-based neural retrieval models to rank video clips in response to a forum discussion. The retrieval models are trained with supervised learning and distant supervision to effectively leverage the unlabeled data—which accounts for more than 80% of all available data. Our experimental results demonstrate that the proposed method is effective for this task, by outperforming a standard baseline by **0.208** on the absolute change in terms of precision.

Keywords

MOOC, Discussion Forum, Video Clip Transcripts, Clip Recommendation

1. INTRODUCTION

Massive Open Online Courses (MOOCs) provide open access to world class courses for the public, which greatly improves the opportunities in online learning. The discussion forum is a major component of a MOOC as it is the primary communication tool among learners and instructors [1] to moderate the lack of physical access in MOOCs. It can help learners build a sense of belonging and learn from peers, or help instructors monitor learner affect and academic progress [2]. However, since questions targeting the same video content are scattered among discussion threads, without supporting

*The first author has been supported by the China Scholarships Council (CSC). The last author has been supported by NWO project SearchX (639.022.722) and NWO Aspasia (015.013.027).

P. Zhu, C. Hauff, and J. Yang. MOOC-Rec: Instructional video clip recommendation for MOOC forum questions. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 705–709, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853055>

navigation facilities, learners cannot effectively retrieve valuable discussions for a particular piece of content. In addition, learners' posts seeking help may be drowned out by the many other competing posts, making it hard for learners to get attention from instructors and peers. The unstructured, unorganized forums with a large amount of discussions (that can lead to information overload [19]) are hindering instructors and learners to benefit from them, decrease community interaction, reduce responsiveness in forums and in the end lead to low MOOC retention rates [20, 13].

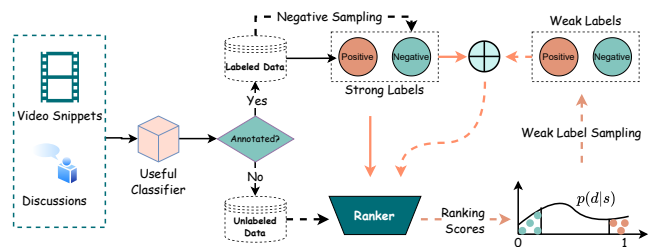


Figure 1: Overview of MOOC-Rec.

Existing works directed at addressing the information overload issue in MOOC forums have proposed more effective navigation tools to identify instructional video contents and make recommendations of a ranked list of video clips. For example, [2] classify posts that need help and employ bag-of-words based retrieval techniques to map those posts to minute-resolution course video clips. The clip recommendation algorithm is evaluated on posts from one course. [17] built a recommender system to generate a ranked list of video clips giving a student's question with a deep neural network; they evaluate the system with 50 questions. Despite these attempts, we argue that prior works on video clip recommendation suffer from a lack of training data, and as a consequence report evaluations only on small-scale data. It remains a challenge to develop and evaluate a system that can scale to thousands of MOOCs, across different domains.

In our work, we first address the lack of training data issue by creating MOOC-CLIP, a novel large-scale dataset from Khan Academy¹, that includes video transcripts and forum posts (both questions and answers) using raw data available from LearningQ [3], an open source tool and dataset for educational question generation. Second, we propose MOOC-Rec,

¹<https://www.khanacademy.org/>

a dense retrieval based instructional video clip recommendation system for MOOC forum questions. For each content-related thread, **MOOC-Rec** recommends a ranked list of video clips that are likely relevant and helpful for answering the question. Although dense retrievers have been applied in various retrieval tasks such as DPR [6] and ColBERT [7], it is unknown whether they are an effective approach for MOOC video clip recommendation. Lastly we point out that only 11.57% of all discussions in our dataset are labeled with a target video clip, which poses challenges for training **MOOC-Rec** with limited labeled data and abundant unlabeled resources.

We here first investigate the effectiveness of **MOOC-Rec** and then we address the scarcity of labeled data by using distant supervision and in-batch negatives to train the ranker. The comprehensive experiments on our large-scale dataset which consists of about 274K discussions show that our systems significantly improve the clip recommendation performance by outperforming a standard baseline by 0.208 in terms of precision.

2. THE MOOC-CLIP DATASET

To address the lack of research data, we create a large-scale dataset using raw data crawled with **LearningQ**² from Khan Academy, a MOOC platform which allows learners to ask and answer questions about the learning materials during learning. We keep video transcripts, forum questions and answers of MOOCs which have both transcripts and discussions available.

Learners use discussion forums in different ways. Besides asking questions related to the course materials, they may also discuss irrelevant topics [14] for the purposes of socializing, spamming, or expressing their appreciation for the course materials. Some questions posted by learners also suffer from a lack of proper context, or are too generic. Therefore, it is necessary to remove these relatively—for our purposes—low-quality questions. In line with **LearningQ**, we consider a user-generated question to be useful for learning when all of the following conditions hold: (i) the question is concept-relevant, i.e., it seeks for information on knowledge concepts taught in lecture videos; (ii) the question is context-complete, containing sufficient context information to enable other learners to answer the question; and (iii) the question is not generic. Besides labeled questions in **LearningQ**, we manually labeled 2K (2,344 in total) as unknown set for our cross-topics evaluation, 8,766 questions on the remaining 5 topics for training, and 2,186 questions as known topic test set. We train a BERT-based text sequence classifier for useful question classification. Table 1 summarizes its performance.

During preprocessing, we first remove noisy discussions which

²<https://github.com/AngusGLChen/LearningQ>

Method	Same Topic			Cross-Topics		
	Acc	Rc	F1	Acc	Rc	F1
Q	89.40	96.68	92.90	77.20	74.49	75.82
Q+C	89.75	96.54	93.02	73.30	82.68	77.71

Table 1: Useful question classifier results.

contain only meaningless tokens, as well as videos which have no discussions. Then we apply the useful question classifier on all items(522K) and retain only items are classified as useful. In the end, we retain 273,887 discussions from 7,349 videos of 6 topics. We use regular expressions to retrieve discussions where learners label posts with exact timestamps in questions or answers. We split the video transcripts into snippets with a one minute length. The discussions and the snippets which cover the timestamp are labeled as *positive* items. The other discussions are treated as unlabeled. Table 2 and Figure 2 summarize the data statistics. In summary, there are 31,680 positive labeled items and 240,551 unlabeled items, i.e. 11.57% of all discussions are labeled.

Split	#V	#S/V	#W/S	#W/Q	#W/A
Train	4590	7.91	198.51	39.96	80.89
Dev	895	8.37	199.04	40.02	79.26
Test	1126	8.14	198.64	39.67	81.92
Unlabeled	7283	7.70	197.96	38.46	78.58

Table 2: Dataset overview, in terms of videos (#V), snippets (#S) per video, discussions (#D) per video, clip (#W), the number of words per question (Q) and the number of words per answer (A)

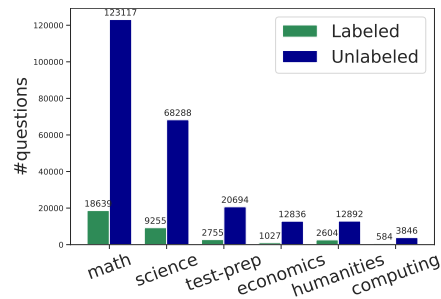


Figure 2: Dataset overview regarding the number of labeled and unlabeled questions in each topic. We can see the unbalanced distribution questions in each topic.

This dataset also covers a series of educational topics including math, science, careers, humanities, etc. We conduct an exploratory analysis along each topic dimension which is shown in Fig 2. We observe a topic imbalance, e.g. discussions under math and science topics account for 78.88% of labeled items and 76.82% of all items. The labeled data is then split into 80% and 20% for training and test sets respectively based on the number of discussions in each set.

3. METHODOLOGY

The problem of MOOC video clip recommendation studied in this paper can be described as follows. Given a forum

discussion question, the system retrieves a ranked list of the most relevant video clips as represented by their transcripts. We assume the questions filtered by the useful question classifier are relevant to the course materials, and the most relevant video clips should be instructional for learners. Assume a MOOC video \mathcal{V} lasts for T seconds, then we split it into s $t = 60$ seconds clips, where $s = \lceil \frac{T}{t} \rceil$. Then the video \mathcal{C} contains clips c_1, c_2, \dots, c_s . Each clip c_i is represented with its transcripts, which can be viewed as a sequence of tokens $w_1^i, w_2^i, \dots, w_{|c_i|}^i$. We also formally define a discussion as $d_i = [q_i, \{a_i\}]$, where $\{a_i\}$ are the answers to the question q_i . Note that in some cases the question has not been answered yet, which is common in MOOC forums. The task is retrieve a ranked list of clips $c_{i,1}, c_{i,2}, \dots, c_{i,s}$ given each discussion d_i . Notice that the video clip recommender needs to work effectively for MOOCs in different domains that the corpus covers. Formally speaking, the recommender $\mathcal{R} : (d, \mathbf{C}) \rightarrow \mathbf{C}_{\mathcal{R}}$ is a function that takes a discussion d and video clip list \mathbf{C} as the input and returns a ranked list of clips $\mathbf{C}_{\mathcal{R}}$. We can also choose to only return the top- K most relevant clips.

3.1 Dual-Encoder

We employ a standard neural IR architecture [6] for the ranker. It uses a dense encoder $E_C(\cdot)$ which encodes the video clip transcripts into m -dimensional real-valued vectors. At run-time, MOOC-Rec maps the input discussion $d = [q, a]$ to another m -dimensional vector using the query encoder $E_Q(\cdot)$, and retrieves the top- k most closest video clip vectors from the same video. We use cosine similarity to model the similarity between the discussion and the clip vectors by the following function:

$$\text{sim}(d, c) = \cos(E_Q(d), E_C(c)). \quad (1)$$

The goal of training is to learn a better embedding function for both the clips and discussions which can map relevant pairs of discussions and clips to vectors with smaller distance, i.e. higher similarity, so that the similarity function $\text{sim}(d, c)$ becomes a good ranking function for the task of MOOC video clip recommendation. This is essentially a *metric learning* problem [9, 11, 6].

Let $\mathcal{M} = \{(d_i, c_i^+, c_{i,1}^-, \dots, c_{i,n}^-)\}_{i=1}^m$ be the training MOOC discussion corpus that contains m instances. Each example has one discussion $d_i = [q_i, a_i]$, one relevant (positive) video clip transcript c_i^+ , and n irrelevant (negative) clips $c_{i,j}^-$. We train the retrieval model by optimizing the negative log likelihood of the positive clip:

$$L(d_i, c_i^+, c_{i,1}^-, \dots, c_{i,n}^-) = -\log \frac{e^{\text{sim}(d_i, c_i^+)}}{e^{\text{sim}(d_i, c_i^+)} + \sum_{j=1}^n e^{\text{sim}(d_i, c_{i,j}^-)}}$$

Positive and Negative Video Clips. For labeled discussions, positive and negative video examples are explicit. We use the video clip whose time duration contains the timestamp of the discussion as the positive example. All other video clips from the same video can be treated as negatives. As MOOC videos vary in the number of clips and to boost the model training and balance the number of positive and negative examples, we selected n of them as the training

negative examples. We apply in-batch negatives [5, 6] for training. In this case, the positive clips for other questions are also treated as the negatives for the current question.

Distant Supervision with Unlabeled Data. As we show in Table 2, over 80% of all discussions are unlabeled (i.e. there is no video timestamp available). It would be labor-intensive and expensive to create human annotations. Thus, we adopt *distant supervision* [10] to effectively utilize the rich unlabeled data and train a better model with them. This process involves training the model with noisy weakly labeled data. MOOC-Rec is able to achieve over 50% precision in top-1 prediction and over 70% in top-3 with a Recall@3 of over 80%. Therefore, we use the ranker trained on the labeled training set as the scorer and clips with the highest $\text{sim}(d, c)$ are selected as positives while the clips with the lowest $\text{sim}(d, c)$ (besides top-3) as negatives. The weakly labeled data are then used to train the ranker.

Inference. During inference time, we pre-compute all clip embedding v_c by applying the clip encoder E_C to all MOOC video clips offline. Given a discussion $d = [q, a]$ at run-time, we concatenate the question and answers if a is available and compute the discussion embedding $v_d = E_Q(d)$. The clips are then ranked by $\text{sim}(d, c)$ and the top- k are retrieved.

Although encoders can be implemented in many different ways [10], in this work, we use two independent BERT [4] variant models as encoders and the mean value of all token embeddings is used as the final representation. We tokenize clip transcripts and truncate the token list to maximum length of 512 (starting with [CLS] and ending with the [SEP] token). The discussion encoder works as a *query* encoder in typical neural IR systems. Instead of using separate encoders for questions and answers of the discussion, in our design both of them share the same encoder. In this way, we train a better query encoder for questions by taking advantage of important answer information.

3.2 Cross-Encoder

Both the cross-encoder and dual-encoder are two common approaches for matching sentence pairs. While the dual-encoder produces sentence embedding vectors for clips and discussions independently, the cross-encoder treats the clip recommendation for discussions as a sequence classification task and performs full self-attention over the entire sequence. We concatenate the video clip transcripts and the discussions (question and answers) with the [SEP] token as the input to the transformer network. The [CLS] token embedding is then passed to a binary classifier to predict the binary relevance between them.

4. EXPERIMENTS AND RESULTS

4.1 Experimental Settings

Implementation. Two BERT variants: MPNet [16] (*abbrv.* MP, embedding size: 768) and MiniLM [18] (*abbrv.* MP, embedding size: 384) are used as text encoders. We implement dual-encoders using pre-trained weights provided by

Sentence-Transformers library³ [15]. Both models are pre-trained on a large and diverse dataset of over 1 billion training query-paragraphs pairs for the semantic search task. The Adam optimizer [8] with warming-up and cosine schedule is used for training; we set the maximum learning rate to $lr = 2e^{-5}$, $\epsilon = 1e^{-8}$ and the warmup steps to 1000. For the **cross-encoder** baseline, we follow previous research [10, 12]. The **BM25** baseline is based on the Okapi BM25 implementation of the `rank_bm25` library⁴. We train our models using 8 GTX-1080 GPUs for 10 iterations with a batch size of 32. As Figure 3 shows, after one iteration, both clip recommendation systems outperform the BM25 baseline.

Table 3: Performance of the proposed MOOC-Rec ranker and baselines on the test set in terms of rank-aware metrics. MLM/MP_{dual} represents the MiniLM or MPNet based dual-encoder and MLM/MP_{cross} represents the MiniLM or MPNet based cross-encoder. “PT” represents ranker performance using pre-trained encoders without fine-tuning. “FT” means fine-tuned model performance. “WL” means the model performance after training with weakly labeled data.

Method	P@1	MRR	MRR@3	nDCG	nDCG@5	
BM25	0.417	0.600	0.550	0.696	0.593	
PT	MLM _{cross}	0.132	0.346	0.254	0.497	0.297
	MLM _{dual}	0.422	0.614	0.568	0.707	0.617
	MP _{cross}	0.135	0.344	0.248	0.495	0.288
	MP _{dual}	0.386	0.583	0.529	0.683	0.576
FT	MLM _{cross}	0.511	0.677	0.641	0.755	0.683
	MLM _{dual}	0.529	0.692	0.658	0.767	0.700
	MP _{cross}	0.613	0.745	0.716	0.807	0.750
	MP _{dual}	0.570	0.720	0.690	0.788	0.730
WL	MLM _{cross}	0.540	0.696	0.661	0.770	0.700
	MLM _{dual}	0.520	0.683	0.646	0.760	0.687
	MP _{cross}	0.625	0.751	0.722	0.812	0.754
	MP _{dual}	0.557	0.711	0.680	0.782	0.720

4.2 Effectiveness of Dense Retrieval

Performance Comparison with Baseline. After several iterations, the models’ performance first improves gradually and then becomes steady as illustrated in Figure 3, which shows the effectiveness of the training system and the effectiveness of the proposed models. Table 3 summarizes the models’ effectiveness on the test set. We use BM25 as our baseline. Sparse vector-space models and the probabilistic BM25 model have been widely used in instructional clip recommendation systems. BM25’s effectiveness in terms of Precision@1 (P@1) and MRR is 0.417 and 0.60 respectively, which shows queries possess more lexical similarity to related MOOC clips than other clips in the course video and BM25 is an effective and strong baseline for this task. First, we find that without fine-tuning, the pre-trained dual-encoder can achieve similar (MPNet), or even better (MiniLM-L6) performance than the BM25 baseline, while the cross-encoders cannot make clip recommendation for discussions without

³<https://github.com/UKPLab/sentence-transformers>

⁴https://github.com/dorianbrown/rank_bm25

training. Second, we observe significant gains ($p = 1.95e^{-7}$) when using the MOOC-Rec neural ranker after it has been trained on the data, with gains of over 0.15 in P@1 and over 0.19 in nDCG scores compared to the BM25 baseline. Thus, dense retrieval is an effective instructional MOOC clip recommendation approach for forum discussions which can model the relevance between discussions and clip transcripts.

Impact of Model Size. To compare the impacts of model size, we use one distilled transformer model MiniLM which contains **22M** parameters and one BERT size model MPNet which contains **109M** parameters. As Table 3 shows, in both cross-encoder and dual-encoder settings, the larger model (i.e. MPNet) achieves better effectiveness after training, which shows that the transformer model with more parameters may have a better potential to model the relevance between clips and discussions.

Comparison of Cross-Encoder and Dual-Encoder. Both cross-encoder and dual-encoder are commonly used for sentence pair matching problems. In Table 3, we observe that with the distilled transformer model the dual-encoder outperforms the cross-encoder by 0.018 in terms P@1. However, with large model, the cross-encoder outperforms dual-encoder by 0.043 on P@1, and around 0.02 on other metrics. Despite the performance advantage of the cross-encoder with a large model, as outlined in Section 3.2, we observe a massive computational overhead with the cross-encoder as illustrated in Figure 5.

Effect of Distant Supervision. In the weakly-labeled data (WL) section of Table 3, we summarize the different models’ performance after distant training with weakly labeled data. Compared with model trained with labeled data only, cross-encoders benefit from WL (+0.029 for MiniLM and +0.012 for MPNet in terms P@1), while dual-encoders perform gets worse (-0.009 for MiniLM and -0.013 for MPNet in terms P@1). Our hypothesis is that although MOOC-Rec achieves a good effectiveness after the initial training, the weakly labeled data created with it still contains considerable noisy content.

5. CONCLUSIONS

We studied the task of video clip recommendation in the context of MOOC forums which has the eventual goal to reduce learners’ information overload. We created a novel dataset MOOC-Clip which includes video transcripts and discussions. We systematically investigated how well the state-of-art pre-trained neural IR models work for the task of MOOC clip recommendation, and proposed a framework including data preparation, useful question classification, clip ranker and weak supervision training for this task. We conducted the experiments with both cross-encoders and dual-encoders. The results on our dataset show that neural IR approaches are indeed effective—at the same time, a P@1 value of less than 0.63 (at best) shows that we are still far away from solving this task. In future work, we plan to further investigate the factors that affect MOOC-Rec’s effectiveness such as the clip duration and methods of creating weak labels.

References

- [1] P. Adamopoulos. What makes a great mooc? an interdisciplinary analysis of student retention in online courses. 2013.
- [2] A. Agrawal, J. Venkatraman, S. Leonard, and A. Paepcke. Youedu: addressing confusion in mooc discussion forums by recommending instructional video clips. 2015.
- [3] G. Chen, J. Yang, C. Hauff, and G.-J. Houben. Learningq: a large-scale dataset for educational question generation. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 12, 2018.
- [4] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [5] M. Henderson, R. Al-Rfou, B. Strope, Y.-H. Sung, L. Lukács, R. Guo, S. Kumar, B. Miklos, and R. Kurzweil. Efficient natural language response suggestion for smart reply. *arXiv preprint arXiv:1705.00652*, 2017.
- [6] V. Karpukhin, B. Oğuz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W.-t. Yih. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*, 2020.
- [7] O. Khatib and M. Zaharia. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 39–48, 2020.
- [8] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [9] B. Kulis et al. Metric learning: A survey. *Foundations and Trends® in Machine Learning*, 5(4):287–364, 2013.
- [10] J. Lin, R. Nogueira, and A. Yates. Pretrained transformers for text ranking: Bert and beyond. *arXiv preprint arXiv:2010.06467*, 2020.
- [11] B. Mitra, N. Craswell, et al. *An introduction to neural information retrieval*. Now Foundations and Trends, 2018.
- [12] R. Nogueira and K. Cho. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*, 2019.
- [13] A. Ntourmas, N. Avouris, S. Daskalaki, and Y. Dimitriadis. Evaluation of a massive online course forum: design issues and their impact on learners’ support. In *IFIP conference on human-computer interaction*, pages 197–206. Springer, 2019.
- [14] A. Ntourmas, S. Daskalaki, Y. Dimitriadis, and N. Avouris. Classifying mooc forum posts using corpora semantic similarities: a study on transferability across different courses. *Neural Computing and Applications*, pages 1–15, 2021.
- [15] N. Reimers and I. Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.
- [16] K. Song, X. Tan, T. Qin, J. Lu, and T.-Y. Liu. MpNet: Masked and permuted pre-training for language understanding. *Advances in Neural Information Processing Systems*, 33:16857–16867, 2020.
- [17] P. Trirat, S. Noree, and M. Y. Yi. Intellimooc: Intelligent online learning framework for mooc platforms. In *Proceedings of The 13th International Conference on Educational Data Mining (EDM 2020)*, pages 682–685, 2020.
- [18] W. Wang, F. Wei, L. Dong, H. Bao, N. Yang, and M. Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in Neural Information Processing Systems*, 33:5776–5788, 2020.
- [19] D. A. Wiley and E. K. Edwards. Online self-organizing social systems: The decentralized future of online learning. *Quarterly review of distance education*, 3(1):33–46, 2002.
- [20] D. Yang, M. Wen, I. Howley, R. Kraut, and C. Rose. Exploring the effect of confusion in discussion forums of massive open online courses. In *Proceedings of the second (2015) ACM conference on learning@ scale*, pages 121–130, 2015.

APPENDIX

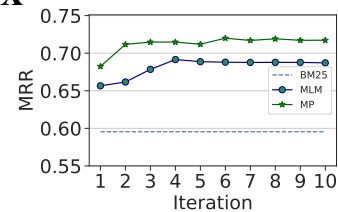


Figure 3: System performance along each training iteration.

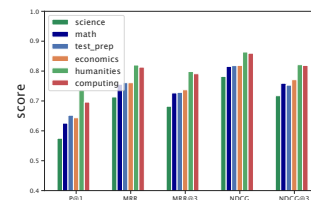


Figure 4: Performance along different topics.

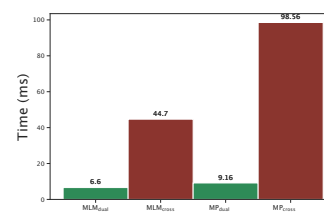


Figure 5: Average Processing Time.

Automatical Graph-based Knowledge Tracing

Ting Long¹, Yunfei Liu¹, Weinan Zhang¹, Wei Xia², Zhicheng He², Ruiming Tang² and Yong Yu¹

¹ Shanghai Jiao Tong University, ² Huawei Noah's Ark Lab
{longting, liuyunfei, wnzhang, yyu}@sjtu.edu.cn, {xiawei24, hezhicheng9, tangruiming}@huawei.com

ABSTRACT

Knowledge tracing (KT) is an essential task in online education, which dynamically assesses students' mastery of concepts by predicting the probability that they correctly answer questions. One of the most effective solutions for knowledge tracing is graph-based methods. They maintain multiple vectors to represent students' mastery of concepts, and use these vectors to predict the probability of students correctly answering questions. To give more accurate predictions, the graph-based methods require concept relations to update these vectors once students answer questions. However, the concept relations usually require manual annotation in a real-world scenario, limiting the application of the graph-based method. In this paper, we proposed a method called Automatical Graph-based Knowledge Tracing (AGKT), which is a graph-based method that updates these vectors without requiring manually annotated concept relations. We evaluate our method on four public datasets and compare it with ten advanced methods. The experiment results demonstrate that AGKT yields superior performance.

Keywords

online learning, knowledge tracing, graph-based method

1. INTRODUCTION

Knowledge tracing is a task that estimates students' mastery of knowledge by predicting the probability that they correctly answer questions. It plays a significantly important role in the online educational application, like exercise recommendation and knowledge diagnosis. The input of knowledge tracing is the question-answering history of a student and a new question, and the output is the probability of the student correctly answering the question. For instance, in step 4, the knowledge tracing aims to use the student's performance on q_1, q_2, q_3 to predict the probability of the student correctly answering q_4 in Figure 1.

T. Long, Y. Liu, W. Zhang, W. Xia, Z. He, R. Tang, and Y. Yu. Automatical graph-based knowledge tracing. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 710–714, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853057>

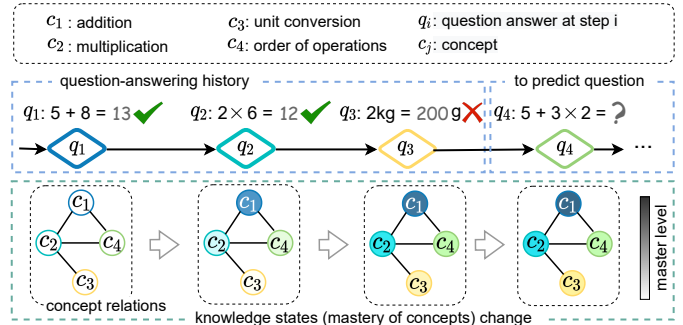


Figure 1: An example of a student's question-answering sequence and the change of her (his) knowledge states. The darkness of a circle's color represents the level of the student masters the corresponding concept, and more darkness denotes a better mastery.

To address the knowledge tracing problem, many outstanding methods [11, 17, 8, 19, 7, 1, 5, 3, 10, 18, 12, 9, 14] have been proposed, which could be grouped into graph-free methods [11, 17, 8, 19, 7, 1, 5, 3, 10, 18, 12] and graph-based methods [9, 14]. Graph-free methods are directly built based on the sequential models, like auto-regressive methods [13, 6, 2], Transformer [15]. They maintain one or multiple vectors to represent students' mastery of knowledge concepts, which denotes as knowledge states, and they predict students' performance based on vectors. As students' knowledge states change with time, to maintain the latest knowledge states for prediction, they update the vectors which represent knowledge states according to the questions and question-concept relation immediately after students answer questions. However, according to the previous research, concepts in one specific domain are correlative with each other [7, 19]. Thus, when a student has a deeper understanding of one concept, her (his) mastery of the correlative concepts also changes. For instance, answering q_1 enhances the student's mastery of *addition* in Figure 1. Since *addition* is correlative with *multiplication* in the concept relation graph, the deeper understanding of *addition* enhances her (his) mastery of *multiplication*. Hence, the knowledge states are not only influenced by questions and question-concept relations, but also influenced by the relations among concepts.

Due to this reason, the graph-based methods introduce the concept-relation graph when they update students' knowl-

edge states. Though the graph-based methods gain many achievements in performance and interpretability, the concept-relation graph usually relies on manual annotation, which is almost impossible for cases with thousands of concepts. To address this issue and keep the superiority of graph-based methods, we proposed a method called Automatic Graph-based Knowledge Tracing (AGKT), which is independent on the manual-annotated concept relation, but could update students' knowledge states like graph-based methods by considering question, question-concept relation and concept-concept relation. We perform experiments on four real-world datasets, and compare our method with ten excellent models. The experiment results reveal that our method has superior performance to the other methods.

2. PROBLEM FORMULATION

Suppose the learning history of a student is $\mathcal{X}_{t-1} = \{(q_1, r_1), (q_2, r_2), \dots, (q_{t-1}, r_{t-1})\}$. Here, q_i denotes the question the student answers at step i . r_i denotes the correctness of the student's response on q_i , and

$$r_i = \begin{cases} 1, & \text{if the student's answer is right;} \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

Each question is related to one or multiple concepts. We denote the set of concepts as $C = \{c_j\}_{j=1}^{|C|}$, and the set of the concepts which are related to q_i as C_i , and the set of the concepts which are unrelated to q_i is denoted as D_i . Obviously, $C_i \cup D_i = C$.

The task of knowledge tracing is formulated as estimating the probability that the student correctly answer a new question q_t given the question-answering history \mathcal{X}_{t-1} , *i.e.*, $P(r_t = 1 | q_t, \mathcal{X}_{t-1})$. We approach that by learning a function to estimate the probability:

$$\bar{r}_t = f_{\Theta}(\cdot), \quad (2)$$

where $\bar{r}_t = P(r_t = 1 | q_t, \mathcal{X}_{t-1})$ and the input of $f_{\Theta}(\cdot)$ represent the features used to predict the correctness of the student.

3. METHOD

As we discussed previously, students' latest knowledge states are necessary to predict the probability of students correctly answering questions. Moreover, considering the question, the question-concept relation and concept-concept relation in knowledge state update benefits for the performance. Nevertheless, annotating the concept-concept relation is almost impossible. To reserve the superiority of the graph-based methods in the condition of manual annotated concept-concept relation absent, we proposed a method called Automatic Graph-based Knowledge Tracing (AGKT). It is composed of three components, as Figure 2 (a) illustrates: Automatic Graph (AG), state update module and prediction module. AG is obtained according to question-concept relation, which replaces the concept relation graph to assist the state update module in updating students' knowledge states. The prediction module predicts the probability of students correctly answering questions by students' knowledge states and the question information. In the following, we will discuss the AG first. Then, we will present how to update students' knowledge states based on AG in the

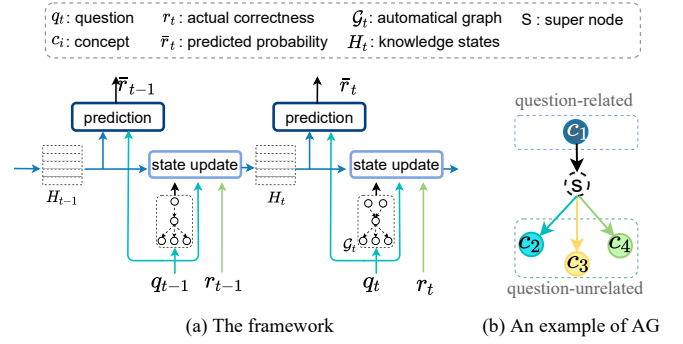


Figure 2: (a) The framework of Automatic Graph-based Knowledge Tracing (AGKT). (b) An example of the Automatic Graph (AG) for the q_1 in Figure 1.

state update module. Finally, we will discuss the prediction module.

3.1 Automatic Graph (AG)

Each question corresponds to an Automatic Graph (AG). Figure 2(b) illustrates the corresponding AG for q_1 in Figure 1. We denote the graph corresponds to q_t as \mathcal{G}_t , and $\mathcal{G}_t = \{\mathcal{V}_t, \mathcal{E}_t\}$. \mathcal{V}_t is the node set, and $\mathcal{V}_t = \{V_t^c, V_t^d, s\}$. Here, each node in V_t^c represents a concept in C_t . That is, each node in V_t^c represents one question-related concept. Each node in V_t^d represents a question-unrelated concept (D_t). For the ease of presentation, we call these nodes by concepts when there is no ambiguity. s is the supernode, which connects the question-related concepts and question-unrelated concepts. \mathcal{E}_t represents the edges on \mathcal{G}_t , and $\mathcal{E}_t = \{E_c, E_d\}$, E_c represents the edges between V_t^c and s , and E_d represents the edges between V_t^d and s .

3.2 The state update module

Since the questions require students to utilize their knowledge of question-related concepts to answer, which increases their comprehension of the question-related concepts, and causes their knowledge states on these concepts to change once they answer questions. As concepts are correlative, these changes will also influence the knowledge states of some question-unrelated concepts. Based on this, we maintain a vector for each concept to represent a student's knowledge state on it. When a student answer a question, we update the knowledge states of question-related concepts first, and then we update the knowledge states of question-unrelated concepts according to the AG.

3.2.1 The state update of question-related concepts

Suppose a student interacts with question q_t at step t . After the student answer q_t , we update the knowledge states of question-related concepts by concepts' attributes and the student's response on q_t . Specifically, for a concept $c_i \in C_t$, we integrate its embedding and the student's correctness by:

$$\mathbf{z}_i^c = \begin{cases} \mathbf{e}_i^c \oplus \mathbf{0}, & r_t = 1, \\ \mathbf{0} \oplus \mathbf{e}_i^c, & r_t = 0, \end{cases} \quad (3)$$

where $\mathbf{0} = (0, 0, \dots, 0)$, is a d dimension zero vector. $\mathbf{e}_i^c \in \mathbf{R}^d$ denotes the embedding of c_i . \oplus denotes the concatenation.

Then, we feed the concatenation into the cell of recurrent neural network (RNN) to update the knowledge state of c_i :

$$\mathbf{h}_{t+1}^i = GRU(\mathbf{z}_t^i, \mathbf{h}_t^i), \quad (4)$$

where \mathbf{h}_t^i denotes the knowledge states on concept c_i at step t , it is the i -th row of the knowledge states matrix \mathbf{H}_t . GRU denotes the gated recurrent unit [2].

3.2.2 The state update of question-unrelated concepts.

The knowledge states of one concept will influence the concepts that correlate to it, *e.g.*, the knowledge state of *multiplication* is influenced by the knowledge state of *addition* after the student answer q_1 in Figure 1. Thus, the update of question-unrelated concepts should be based on the relation of concepts. Since the concept relation is unknown in many cases, we update the knowledge states of question-unrelated concepts based on AG (Figure 2 (b)). Since the question-related concepts connect to the supernode, and the supernode connects the question-unrelated concepts in AG, the supernode integrates the message about the knowledge states of all the question-related concepts first. Then it transmits the integrated message to question-unrelated concepts. Finally, the question-unrelated concepts update their knowledge states according to the message transmitted from supernode. Specifically, we take the following steps to update the knowledge states of the question-unrelated concepts:

First, we represent the original message that a question-related concept c_i sent to the supernode by:

$$\mathbf{m}_t^i = \mathbf{h}_{t+1}^i \oplus \mathbf{o}_i, \quad (5)$$

where \mathbf{o}_i is the one-hot encoding of concept c_i to denote the identity of the message source. The supernode aggregates the messages from the question-related concepts by:

$$\mathbf{h}^s = \frac{1}{|C_t|} \sum_{c_i \in C_t} f_s(\mathbf{m}_t^i), \quad (6)$$

where f_s denotes multi-layer perceptrons (MLP). \mathbf{h}^s denotes the states of the supernode.

Then, the supernode transmits the message it receives to the question-unrelated concepts. We represent the message obtained by the question-unrelated concept c_j ($c_j \in D_t$) as

$$\mathbf{m}_t^j = f_{d,j}(\mathbf{h}^s), \quad (7)$$

$f_{d,j}$ denotes the mapping function, which is implemented by MLP, to extract the message which is interested by c_j , *e.g.*, the knowledge states of its correlative concepts. Note, for different question-unrelated concepts, the mapping functions in Eq. 7 are different. However, for the same question-unrelated concept in different time steps, the mapping function is shared to guarantee the pattern are compatible in all steps.

Finally, we update the knowledge state on question-unrelated concept c_j by:

$$\mathbf{h}_{t+1}^j = GRU(\mathbf{m}_t^j, \mathbf{h}_t^j). \quad (8)$$

3.3 The prediction module

To predict the probability of students correctly answering questions, we consider the attributes of question q_t , the level

Table 1: Dataset statistics.

Dataset	ASSIST09	ASSIST12	EdNet	Junyi Math
Students	2,968	22,422	50,000	1,146
Records	185,110	1,839,429	3,266,010	101,854
Questions	15,003	45,543	12,077	1,145
Concepts	121	99	189	39
Questions Per Concept	150.76	460.03	144.01	36.28
Concepts Per Question	1.22	1.0	2.67	1.0
Attempts Per Question	12.34	40.39	270.43	71.98
Attempts Per Concept	1,914.21	18,580.10	39,959.14	2,611.64
Positive Label Rates	63.80%	69.60%	59.54%	66.78%

of the student master question-related concepts, and her(his) knowledge background. We represent the attributes of question q_t by the question embedding \mathbf{e}_t^q , and $\mathbf{e}_t^q \in \mathbf{R}^d$. We represent the level of the student master the question-related concepts by the mean knowledge state:

$$\mathbf{h}_m = \frac{1}{|C_t|} \sum_{c_i \in C_t} \mathbf{h}_t^i. \quad (9)$$

We represent students' knowledge background as

$$\mathbf{h}_c = \frac{1}{|C|} \sum_{c_i \in C} \mathbf{h}_t^i, \quad (10)$$

and then we predict the probability of the student correctly answering q_t by

$$\bar{r}_t = \delta(f_r(\mathbf{h}_c \oplus \mathbf{e}_t^q \oplus \mathbf{h}_m)), \quad (11)$$

where f_r denotes MLP, and δ denotes the Sigmoid function.

3.4 Model Learning

The objective function of our model is to minimize the negative log-likelihood of the observed sequence. The sequence is the question-answering history of the student from step 1 to T . The learning parameters of our method are the embedding of concepts and questions, the weights in GRU, the parameters of all the MLPs. The parameters are jointly learned by minimizing the cross-entropy between the predicted probability \bar{r}_t and the students' actual correctness r_t as

$$\mathcal{L} = - \sum_{i=1}^T (r_i \log \bar{r}_i + (1 - r_i) \log(1 - \bar{r}_i)). \quad (12)$$

4. EXPERIMENT

4.1 Dataset

We evaluate our method on four public datasets: ASSIST09, ASSIST12, EdNet, and Junyi Math. These datasets record the question-answering history of students. We take the questions, the concepts related to the questions, and the students correctness of responses from the records. The maximum length of students' question-answering history is set to 200. We split 80% data for training and validation, and 20% for testing. The statistics of the four datasets are shown in Table 1. Note, the statistics are the actual samples we use in our experiments after preprocessing, which are different from the statistics of raw data.

4.2 Baselines

To evaluate the effectiveness of our model, we compare our method with graph-free methods, and graph-based methods. The graph-free methods trace students' knowledge

Table 2: The AUC on four public datasets.

Model	ASSIST09	ASSIST12	EdNet	Junyi Math
BKT	0.6815	0.6142	0.5729	0.6293
KTM	0.6734	0.6881	0.7071	0.7207
SAKT	0.6884	0.6914	0.7313	0.7422
SAINT	0.6901	0.6917	0.7336	0.8079
AKT-NR	0.6940	<u>0.7098</u>	0.7450	0.7705
DKT	0.6769	0.6884	0.7496	0.8397
DHKT	<u>0.7499</u>	0.6966	<u>0.7547</u>	<u>0.8594</u>
EERNNA	0.7244	0.7000	0.7437	0.8196
DKVMN	0.7235	0.6664	0.7009	0.8426
GKT	0.7488	0.6857	0.7039	0.8257
AGKT	0.7765	0.7232	0.7589	0.8656

states without considering concept-relation, and they could be grouped into four groups. The first group are traditional methods, *i.e.*, BKT [4], KTM [16], which trace the students’ knowledge states according to the factors that affect students’ learning. The second group are single-state methods, *i.e.*, DKT [11], DHKT [17], EERNNA [7], which use one vector to represent students’ knowledge states on all concepts. The third group is multi-state methods, *i.e.*, DKVMN [19], which maintain a vector for each concept to represent students’ knowledge states on them. The fourth group are state-free methods, *i.e.*, SAKT [10], SAINT [3], AKT-NR [5], which maintain no explicit vector to represent students’ knowledge states. The graph-based methods apply a concept-relation graph in the knowledge state update. Here we choose the GKT [9].

4.3 Student Response Prediction

We measure the AUC to evaluate the performance of models. A higher AUC indicates a better performance. Table 2 shows the converged ACC, AUC. According to Table 2, on the ASSIST09, our method is better than the best baseline DHKT by 2.66% in AUC. On ASSIST12, our model outperforms the best baseline AKT-NR 1.34% in AUC. On EdNet, our model is better than the best baseline DHKT by 0.42% in AUC. On Junyi, our method is better than the best baseline DHKT 0.64% in AUC. Thus, the performance presented in Table 2 demonstrates that our method is effective.

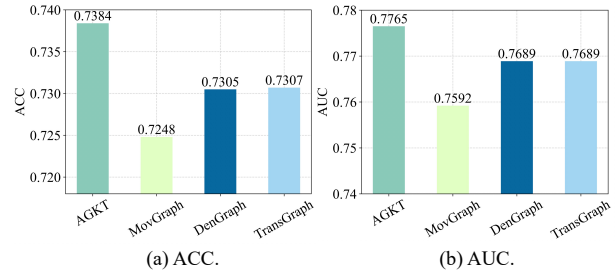
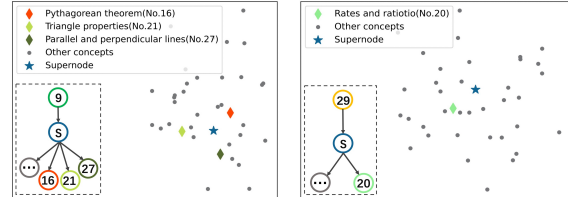
4.4 Ablation Study

To further verify the contribution of AG, we conduct experiments on ASSIST09 with three comparative settings:

- **MovGraph** removes the AG from AGKT. Thus, this setting removes the concept-concept relation.
- **DenGraph** replaces the AG in AGKT with the dense graph in GKT [9]
- **TransGraph** replaces AG with the transition graph in GKT.

The experimental results are shown in Figure 3. We can find that: (1) *AGKT*, *DenGraph* and *TransGraph* have better performance than *MovGraph*. That means the consideration of concept-concept relation is beneficial; (2) Replacing AG in AGKT with other graphs (*DenGraph*, *TransGraph*) decreases the performance. That means the AG is more effective in the framework of AGKT.

4.5 Case Study

**Figure 3: The contribution of AG.****Figure 4: The state of supernode and the message received by the question-unrelated concepts.****Figure 4: The state of supernode and the message received by the question-unrelated concepts.**

To investigate whether the states update of question-unrelated concepts is interpretable, we randomly pick two question-answering records, and visualize the state of supernode and the message obtained by question-unrelated concepts. We expect the knowledge states of question-unrelated concepts could be automatically updated according to the actual relationship with the question-related concepts when AGKT converges. Thus, for the correlative concepts of the question-related concepts, the message they receive from supernode should be correlated with the state of the supernode.

The results is presented in Figure 4. We can observe the message obtained by *triangle properties* (No.21) is highly close to supernode when the student answer the question related to *congruent triangles* (No.9), and message obtained by *rates and ratios* is highly close to the state of supernode when the student answer the question related to *ratio-percentage* (No.29). Thus, the state update of question-unrelated concepts is interpretable in our method. which demonstrates our method has good interpretability.

5. CONCLUSION

In this paper, we proposed a method called Automatic Graph-based Knowledge Tracing (AGKT). Different from the previous graph-based methods, it adopts the Automatic Graph (AG) to automatically update students’ knowledge state without requiring the manually annotated concept relation. We evaluate the performance of our method on four public real-world datasets, and compare it with ten methods. The experiment result reveals that our method is effective in tracing student’s knowledge states.

6. ACKNOWLEDGMENTS

The corresponding author Yong Yu is supported by National Natural Science Foundation of China (62177033). The work is also sponsored by Huawei Innovation Research Program.

References

- [1] G. Abdelrahman and Q. Wang. Knowledge tracing with sequential key-value memory networks. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 175–184, 2019.
- [2] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.
- [3] Y. Choi, Y. Lee, J. Cho, J. Baek, B. Kim, Y. Cha, D. Shin, C. Bae, and J. Heo. Towards an appropriate query, key, and value computation for knowledge tracing. In *Proceedings of the Seventh ACM Conference on Learning@ Scale*, pages 341–344, 2020.
- [4] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4(4):253–278, 1994.
- [5] A. Ghosh, N. Heffernan, and A. S. Lan. Context-aware attentive knowledge tracing. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2330–2339, 2020.
- [6] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [7] Q. Liu, Z. Huang, Y. Yin, E. Chen, H. Xiong, Y. Su, and G. Hu. Ekt: Exercise-aware knowledge tracing for student performance prediction. *IEEE Transactions on Knowledge and Data Engineering*, 33(1):100–115, 2019.
- [8] K. Nagatani, Q. Zhang, M. Sato, Y.-Y. Chen, F. Chen, and T. Ohkuma. Augmenting knowledge tracing by considering forgetting behavior. In *The World Wide Web Conference*, pages 3101–3107, 2019.
- [9] H. Nakagawa, Y. Iwasawa, and Y. Matsuo. Graph-based knowledge tracing: Modeling student proficiency using graph neural network. In *2019 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*, pages 156–163. IEEE, 2019.
- [10] S. Pandey and G. Karypis. A self-attentive model for knowledge tracing. *arXiv preprint arXiv:1907.06837*, 2019.
- [11] C. Piech, J. Bassen, J. Huang, S. Ganguli, M. Sahami, L. J. Guibas, and J. Sohl-Dickstein. Deep knowledge tracing. volume 28, pages 505–513, 2015.
- [12] S. Shen, Q. Liu, E. Chen, H. Wu, Z. Huang, W. Zhao, Y. Su, H. Ma, and S. Wang. Convolutional knowledge tracing: Modeling individualization in student learning process. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1857–1860, 2020.
- [13] S. Sukhbaatar, A. Szlam, J. Weston, and R. Fergus. End-to-end memory networks. *arXiv preprint arXiv:1503.08895*, 2015.
- [14] S. Tong, Q. Liu, W. Huang, Z. Huang, E. Chen, C. Liu, H. Ma, and S. Wang. Structure-based knowledge tracing: An influence propagation view. In *2020 IEEE International Conference on Data Mining (ICDM)*, pages 541–550. IEEE, 2020.
- [15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.
- [16] J.-J. Vie and H. Kashima. Knowledge tracing machines: Factorization machines for knowledge tracing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 750–757, 2019.
- [17] T. Wang, F. Ma, and J. Gao. Deep hierarchical knowledge tracing. In *12th International Conference on Educational Data Mining, EDM 2019*, pages 671–674. International Educational Data Mining Society, 2019.
- [18] S. Yang, M. Zhu, J. Hou, and X. Lu. Deep knowledge tracing with convolutions. *arXiv preprint arXiv:2008.01169*, 2020.
- [19] J. Zhang, X. Shi, I. King, and D.-Y. Yeung. Dynamic key-value memory networks for knowledge tracing. In *Proceedings of the 26th international conference on World Wide Web*, pages 765–774, 2017.

Process-BERT: A Framework for Representation Learning on Educational Process Data

Alexander Scarlatos¹, Christopher Brinton², and Andrew Lan¹

University of Massachusetts Amherst¹, Purdue University²

ajscarlatos@cs.umass.edu, cgb@purdue.edu, andrewlan@cs.umass.edu

ABSTRACT

Educational process data, i.e., logs of detailed student activities in computerized or online learning platforms, has the potential to offer deep insights into how students learn. One can use process data for many downstream tasks such as learning outcome prediction and automatically delivering personalized intervention. In this paper, we propose a framework for learning representations of educational process data that is applicable across different learning scenarios. Our framework consists of a pre-training step that uses BERT-type objectives to learn representations from sequential process data and a fine-tuning step that further adjusts these representations on downstream prediction tasks. We apply our framework to the 2019 nation’s report card data mining competition dataset that consists of student problem-solving process data and detail the specific models we use in this scenario. We conduct both quantitative and qualitative experiments to show that our framework results in process data representations that are both predictive and informative.¹

Keywords

Process data, representation learning, transfer learning

1. INTRODUCTION

Student modeling [14] is a key research area in educational data mining since it produces estimates of individual factors that affect learning outcomes, including knowledge factors and psychosocial factors such as affect and interest, and informs personalization. There exist a wide range of student models, from those that analyze student *responses* to questions, such as item response theory [12] and models for knowledge tracing [3], to those that analyze student *activity* within digital learning platforms [1, 15, 16].

Educational process data, i.e., data that logs detailed student activity in digitized learning/testing environments, offers us an opportunity to look deeper into the process of learning for each individual student. One can use this process data in many ways: First, standalone process data, especially data from intelligent tutoring systems, learning management systems, or massive open online courses A. Scarlatos, C. Brinton, and A. Lan. Process-BERT: A framework for representation learning on educational process data. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 715–719, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.

<https://doi.org/10.5281/zenodo.6853006>

(MOOCs), can help us capture student behavioral patterns and predict future learning outcomes [9] or help prevent early dropout [6]. Second, process data during assessments, such as the dataset used in the 2019 nation’s report card (NAEP) data mining competition [11], can help us reconstruct the exact process behind how students construct their response to a question. This reconstructed process can potentially help us improve our estimate of student knowledge levels more than using only observed response [2].

One key challenge in educational process data analysis is how to represent process data. Creating good representations is key to improving performance in downstream tasks, as evident in recent advances in other fields, such as pre-trained language models like BERT [4] in natural language processing. The idea is simple: since we often lack a large amount of labels on the variable of interest in the prediction task, e.g., student learning outcomes, using these labels to learn representations of student process data in a supervised learning setup is insufficient and can lead to overfitting. Instead, we start with using the rich process data itself in a *pre-training* step to learn representations through self-supervised learning before *fine-tuning* these representations in the actual downstream prediction task.

1.1 Contributions

In this paper, we propose a generic framework for representation learning from educational process data and apply it to the NAEP Competition dataset [11]. First, we detail how to learn process data representations in a pre-training setup using objectives similar to those used in BERT. We then detail how to fine-tune these representations and use them in a downstream supervised learning task, e.g., predicting learning outcomes. Second, we apply our framework to problem-solving clickstreams as students take an online NAEP assessment and detail our modeling designs. Third, we conduct quantitative experiments to show that our framework is competitive with existing methods in multiple learning outcome prediction tasks. We also conduct qualitative experiments to show that our framework is able to learn meaningful process data representations.

2. METHODOLOGY

In this section we describe our framework, as well as apply it to several learning outcome prediction tasks using the NAEP 2019 competition dataset [11]. The basic ideas behind our framework follow from those in natural language processing (NLP) research but are adapted for student learning process

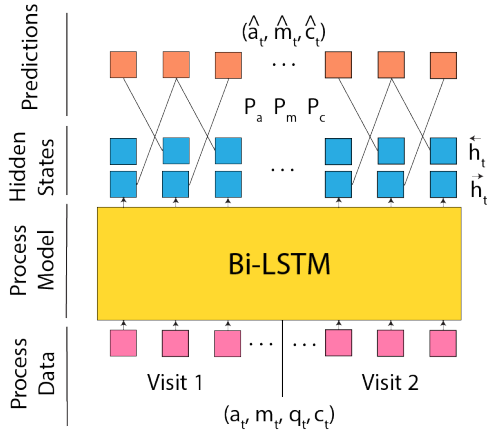


Figure 1: The process model is pre-trained by predicting the properties of each event given surrounding context.

data. There are three main technical components in our framework: 1) the **process model**, which takes a student’s process data as input and produces its *latent representation vectors* as output, 2) the **pre-training objectives**, which are a series of prediction tasks that we use in a self-supervised pre-training phase on the process data to learn its representations via the process model, and 3) the **transfer function**, which adapts the output of the process model, i.e., the latent representations of process data, for use in a downstream learning outcome prediction task.

2.1 Dataset

The dataset contains clickstream logs from students working on two blocks of online NAEP assessments, referred to as blocks A and B. Students are given a time limit of 30 minutes per block, in which they can complete questions in any order. The questions vary in type, including multiple choice, matching, fill in the blank, and “mixed” types. Each event in the log represents a single student action, such as selecting an option in a multiple choice question, typing a character into an answer field, opening the calculator tool, etc. Each raw event in the log contains the student ID, question ID, question type, event type, timestamp, and possibly additional event-specific information such as the target of a click event or the field and character of a key press event. To apply our framework to the NAEP assessment scenario, we process each event so that it has the form $e_t = (a_t, m_t, q_t, c_t)$, where a_t is the event type, m_t is the number of seconds since the student started the test, q_t is an identifier for the current question, and c_t is a response status, which is either *correct*, *incorrect*, or *incomplete*. We additionally define a *visit* as a contiguous sequence of events that are part of the same question, as students may return to previously visited questions within a block. The sequences we provide to the process model each comprise all events of a single student working on a single question across all visits to that question.

2.2 Process Model and Pre-training

We now detail our process model, as well as the objectives used to pre-train it before it can be used on downstream tasks. Our process model is a Bi-LSTM [7], which contains two LSTM’s that run in parallel; one processes the sequence

in order, and the other processes it in reverse. The hidden state of the forward LSTM at time step t , \vec{h}_t , contextualizes e_0, \dots, e_t . Similarly, the hidden state of the backward LSTM at time step t , \overleftarrow{h}_t , contextualizes e_t, \dots, e_T , for sequences of length T . As input to the model, we use a vectorized form of each event, e_t , which contains the concatenation of m_t , learnable embeddings of a_t and q_t , and a one-hot encoding of c_t .

To pre-train the model, we design separate objectives to predict the *event type*, *timestamp*, and *response status* of each event in a student’s process data for a question. By learning to predict these features of the input using context, as we will show, the model becomes able to make inferences about the data, which will be useful for making predictions in downstream tasks. Note that we don’t include a pre-training objective for q_t because it is the same for every event in a question. The flow of data from input sequence to prediction is shown in Figure 1.

We predict the properties of event e_t at each time step by leveraging its full context, which includes all preceding and following events, i.e., $\{e_1, \dots, e_{t-1}, e_{t+1}, \dots, e_T\}$. Since \vec{h}_{t-1} uses on information from $\{e_1, \dots, e_{t-1}\}$ and \overleftarrow{h}_{t+1} uses on information from $\{e_{t+1}, \dots, e_T\}$, we use $\mathbf{z}_t = (\vec{h}_{t-1}, \overleftarrow{h}_{t+1})$ as the encoded context that will be used for prediction. We predict a_t by passing \mathbf{z}_t through a linear prediction head, P_a , using the softmax function [5] to get a probability distribution over possible event types, and use cross-entropy to calculate the loss. We predict c_t in the same way. We design a similar objective for m_t , except that we predict an alternate target, $r_t = \frac{m_t - m_{t-1}}{m_{t+1} - m_{t-1}}$, which represents the portion of time between the prior and following events when m_t occurs. This value is bounded by 0 and 1, which allows us to use binary cross-entropy as the loss function. The advantage to this method over mean-squared error is that because time lapses between student events can vary greatly, the loss for each event is treated more equally. The final pre-training loss for the sequence, \mathcal{L}_{PT} , is the sum of the prediction loss for a_t , m_t , and c_t for each event in the sequence. My minimizing \mathcal{L}_{PT} , the process model learns to reconstruct events based on surrounding context in a self-supervised setup, thus encoding relevant information in its latent states as a result.

2.3 Transfer Learning

We now detail a transfer function, Q_ϕ , that produces a fixed-size output for a downstream prediction task, given the latent states of the process model as input. For the purpose of transfer learning, we define the outputs of the process model to be $\mathbf{z}_1, \dots, \mathbf{z}_T = ((\vec{h}_1, \overleftarrow{h}_1), \dots, (\vec{h}_T, \overleftarrow{h}_T))$. This setup results in each \mathbf{z}_t containing contextualized information that is relevant to the input at time step t , as was ensured by the pre-training process. We combine these outputs using a learnable attention module [5], which assigns a weight to each output latent state of the process model, and then uses the weights to combine all outputs into a single vector. We define the weight vector as $\mathbf{w} = \text{softmax}(\phi_w(\mathbf{z}_1), \dots, \phi_w(\mathbf{z}_T))$, where ϕ_w is a learnable linear projection. We then generate a single vector to represent the entire sequence: $\mathbf{b} = \sum_{t \in \{1, \dots, T\}} w_t \cdot \mathbf{z}_t$.

We can now use this representative vector in a several ways to solve downstream tasks. In the simple case, where we predict the learning outcome at the level of a single question, e.g., the correctness of the student’s response to the question, we can generate the prediction $\hat{y} = \phi_q(\mathbf{b})$, where ϕ_q is a learnable linear projection. Additionally, we may want to predict the learning outcome at the level of a student, e.g., the student’s score on a future test, which requires us to combine the representative vectors for each question in the student’s process data. To achieve this, we define a feed-forward neural network ϕ_s , and generate predictions using the concatenation of the representative vectors for each question: $\hat{y} = \phi_s([\mathbf{b}_1, \dots, \mathbf{b}_Q])$, where \mathbf{b}_j corresponds to the representation of the student’s activity on question j and Q is the total number of questions. In either case, each ϕ in the transfer function can be trained by using an appropriate loss function on the predicted label. In the next section, we show an example of how the transfer function can be used to enhance a separate prediction task.

2.4 Item Response Theory

A popular framework for making predictions at the question level is item response theory (IRT) [8]; we now detail how to enhance it with behavioral data. The 1PL version of IRT learns an *ability* value for each student and a *difficulty* value for each question, which are used to predict the probability that a student will answer a question correctly. We introduce a new *behavior* term, which will adjust the model’s prediction based on the student’s process data for the question. We derive the term using the transfer function: $B_{ij} = \phi_q(\mathbf{b}_j^i)$, where \mathbf{b}_j^i is the representative vector for student i ’s process data on question j , as generated by the process model and the attention module. The predicted probability of a student getting a question correct is defined as $P(Y_{ij} = 1) = \sigma(k_i - d_j + B_{ij})$, where Y_{ij} is an indicator of the correctness of the student’s response, k_i is the learned student ability, d_j is the learned question difficulty, and σ is the sigmoid function. By minimizing the binary cross-entropy loss of the prediction with the actual correctness label, the model can jointly learn \mathbf{k} , \mathbf{d} , and Q_ϕ . Note that for this task, we must remove all indications of response status from the data and not perform the response status pre-training objective, since including them would leak information that could infer the label directly.

3. EXPERIMENTS

In this section, we present the experimental results of our framework applied to the NAEP competition data on two learning outcome prediction tasks. We compare our framework to existing baselines for both the NAEP process data and other process datasets. We finally investigate the interpretability of the process model’s latent representations using visualizations and qualitative analysis.

3.1 Baselines

We use two baselines to compare our framework against. The first is **Feature Engineering (FE)**, for which we used the method of the 2nd place submission from the NAEP 2019 competition [10]. Their technique calculates a large number of features for each student activity sequence, uses a genetic algorithm (GA) to select the best set of features for a target prediction label, and then trains a model ensemble to

Table 1: We report the AUC of predictions on the score label, as well as the AUC when response status is removed.

Model	Test AUC	Test AUC without c_t
FE	0.828	–
CKT	0.854 ± 0.005	0.797 ± 0.010
Ours	0.868 ± 0.008	0.792 ± 0.004

produce a final prediction for the task. We re-ran their GA and ensemble algorithms to obtain predictions for our per-student prediction label. The second baseline is **Clickstream Knowledge Tracing (CKT)**, for which we adapted the technique developed in [2], which uses student problem solving process data for the downstream task of knowledge tracing (KT). Their technique uses an autoencoder pre-training setup where question-level process data is encoded, passed through a bottleneck vector, and reconstructed. We use the bottleneck vector as input to downstream prediction tasks.

3.2 Per-Student Label

We first evaluate our method on a per-student label, referred to as the *score label*, which is a binary indicator of if a student scored above or below average in the second block (B) of the exam. We predict this label by observing process data exclusively from the first block (A).

Experimental Setup. For our model and the CKT baseline, we perform a multi-phase cross-validation experiment. For each fold, using the questions in the training split, we pre-train the process model, and then train the transfer function on the score label. For both of these, we use the validation split for early stopping. Finally, we evaluate the model on the test set, which is fixed across folds. We measure area under the receiver operating characteristic curve (AUC), and report the average and standard deviation of the AUC on the test set over all folds. Note that the FE baseline only returns a single set of predictions for the ensemble.

Results and Discussion. As shown in the second column of Table 1, both CKT and our model outperformed the FE baseline. This observation fits our expectation: since sequential neural models have direct access to the raw process data, they are able to pick up on subtleties that may not be captured by human-engineered features. Our method slightly outperforms CKT on this label, indicating that our process model and transfer function are able to capture more information that is indicative of student performance than CKT. We also examine the ability of the models to predict performance strictly using behavioral information, without any indication of correctness. To do this, we repeat the experiments but remove the response status c_t from the input and do not perform the associated pre-training objective. We see from the third column of Table 1 that the AUC drops, as expected, but is still considerably high, indicating that these models can infer student performance from their behavior.

3.3 Item Response Theory

We now evaluate our model in the IRT setting in order to examine if our methodology can improve performance predic-

Table 2: We report the AUC of predictions on the IRT task, as well as the AUC when incomplete questions are not considered.

Model	All Questions	Completed Questions
Base	0.824 ± 0.001	0.823 ± 0.001
CKT	0.836 ± 0.000	0.830 ± 0.001
Ours	0.836 ± 0.002	0.828 ± 0.001

tion on questions using additional data on student behavior within a question.

Experimental Setup. For the IRT experiments, we use all student activity sequences across blocks A and B. We split the questions into train/test sets using multi-label stratification to ensure all ability and difficulty parameters are sufficiently trained. We then perform a similar cross-validation experiment to the score label, where for each fold, the process model is pre-trained, and then the transfer function and IRT-specific parameters are trained on the question correctness label. The AUC on the test set is recorded after each fold, and the average and standard deviation are reported.

Results and Discussion. We see from Table 2 that the behavioral data leveraged by both our model and CKT result in a small improvement in test AUC over the base IRT model. We also observe that questions that were left incomplete by students were very easy to predict as incorrect with process data since certain event types missing in a student activity sequence clearly indicate incomplete status. To account for this observation, we also report the AUC after removing incomplete questions from the test set. We see that the performance drops for all models, although more significantly for the behavior-enhanced models, leaving our method’s performance slightly below CKT. However, the fact that the behavior-enhanced models still improve over the base IRT model suggests that student behavior provides important additional information on student performance beyond the original student ability and question difficulty parameters in IRT.

3.4 Qualitative Analysis

We now examine the interpretability of the latent behavioral vectors that our methodology produces. We will examine student-level latent vectors extracted from a version of the student-level model, which was modified to capture task-switching behavior. We also performed an analysis of question-level latent vectors extracted from the behavior-enhanced IRT model, where we saw strong behavior-based clustering patterns. However, because question-level behavior representation is similar to the capabilities of CKT, we omit this analysis from this paper, and its details can be found in the long version. For the following figure, we use t-SNE [13] to visualize the latent vectors in 2D and investigate characteristic behavioral patterns in the visible clusters.

To investigate if our model can capture high-level task-switching and test-taking behaviors, we investigate student-level representations, which combine all question visits of a student into a single latent vector. We train a model on

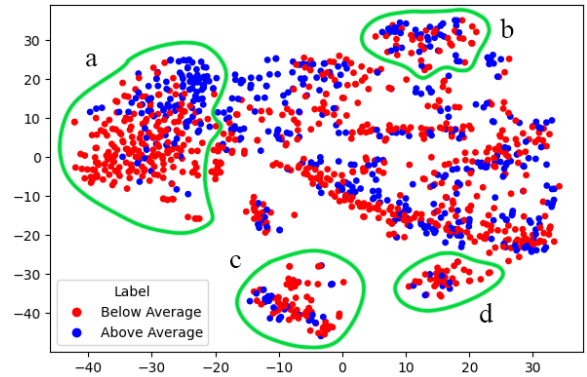


Figure 2: The vectors extracted from the final hidden state of an RNN that processes all visit-level vectors from a student’s process data in block A, colored by the score label.

the score label task with the following modifications to our original setup: 1) we provide the process model with sequences of events in a single visit to a question, rather than across all visits to a question, 2) we add a new pre-training objective to predict the question ID of each event to encode task-switching information in the latent representations, and 3) we replace the fully-connected neural network with an RNN, since the visits are sequential in nature. We use the final hidden state of the RNN for both label prediction and representation visualization.

In Figure 2, we visualize the student-level vectors generated from block A data, colored according to the score label. We identify 4 distinct clusters, while the rest of the vectors have no obvious pattern: a) Rapid testing: most students in this group finished all questions in block A with a significant amount of time remaining. b) Checked their work: students in this group made multiple visits to most questions, often not making changes in the second visit. c and d) Ran out of time: both of these clusters represent students that took the whole time but did not answer all the questions.

4. CONCLUSIONS

In this paper, we developed a BERT-style framework for pre-training and transfer learning on educational process data. We applied our framework to several downstream learning outcome prediction tasks on NAEP assessment process data used in the NAEP 2019 data mining competition. Through quantitative and qualitative experiments, we demonstrated that models developed with our framework can 1) leverage process data to make accurate learning outcome predictions, and 2) generate meaningful representations of student behavior from process data. There are several potential areas for future research. First, our process model currently only represents single questions, which results in student-level representations that are less meaningful than the question-level representations. Future work should aim to develop a process model that can simultaneously represent events across all questions in a student’s process data. Second, future work should aim to implement this framework on complementary datasets, such as video clickstreams, to validate its ability to capture behavioral data across settings.

5. REFERENCES

- [1] A. F. Botelho, R. S. Baker, and N. T. Heffernan. Improving sensor-free affect detection using deep learning. In E. André, R. Baker, X. Hu, M. M. T. Rodrigo, and B. du Boulay, editors, *Int. Conf. Artif. Intell. Edu.*, pages 40–51. Springer International Publishing, 2017.
- [2] W.-L. Chan and D.-Y. Yeung. Clickstream knowledge tracing: Modeling how students answer interactive online questions. In *Int. Learn. Analytics and Knowl. Conf.*, pages 99–109. Association for Computing Machinery, 2021.
- [3] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4(4):253–278, 1994.
- [4] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [5] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.
- [6] S. Halawa, D. Greene, and J. Mitchell. Dropout prediction in moocs using learner activity features. *Proceedings of the second European MOOC stakeholder summit*, 37(1):58–65, 2014.
- [7] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, Nov. 1997.
- [8] F. Lord. *Applications of Item Response Theory to Practical Testing Problems*. Erlbaum Associates, 1980.
- [9] Z. A. Pardos, R. S. Baker, M. O. San Pedro, S. M. Gowda, and S. M. Gowda. Affective states and state tests: Investigating how affect and engagement during the school year predict end-of-year learning outcomes. *Journal of Learning Analytics*, 1(1):107–128, 2014.
- [10] N. Patel, A. Sharma, T. Shah, D. Lomas, et al. Modeling naep test-taking behavior using educational process analysis. *Journal of Educational Data Mining*, 13(2):16–54, 2021.
- [11] T. Patikorn, N. Heffernan, R. Baker, B. Woolf, I. Kats, C. Forsyth, and J. Ocumpaugh. Nation’s report card data mining competition 2019. online: <https://sites.google.com/view/dataminingcompetition2019/home>.
- [12] W. J. van der Linden and R. K. Hambleton. *Handbook of modern item response theory*. Springer Science and Business Media, 2013.
- [13] L. Van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11):2579–2605, 2008.
- [14] K. VanLehn. Student modeling. *Found. Intell. Tut. Syst.*, pages 55–78, 1988.
- [15] D. Yang, T. Sinha, D. Adamson, and C. P. Rosé. Turn on, tune in, drop out: Anticipating student dropouts in massive open online courses. In *Data-driven Educ. WRKSH Conf. Neural Inf. Process. Syst.*, volume 11, page 14, 2013.
- [16] M. Yao, S. Zhao, S. Sahebi, and R. Feyzi Behnagh. Stimuli-sensitive hawkes processes for personalized student procrastination modeling. In *The World Wide Web Conf.*, pages 1562–1573. Association for Computing Machinery, 2021.

Online Item Response Theory (OIRT) - Tracking Student Abilities in Online Learning System

Luyao Peng

ByteDance

pengluyao.phd@bytedance.com

Chengzhi Wei

ByteDance

weichengzhi.franz@bytedance.com

ABSTRACT

In this study, we proposed an Online Item Response Theory Model (OIRT) by combining the Item Response Theory and Performance Factor Analysis (PFA) models. We fitted the proposed model with modified Variational Inference (VI) to perform real-time student and item parameter estimation using both simulated data and real time series data collected from an online adaptive learning environment. Results showed that modified VI parameter estimation method outperformed other Bayesian parameter estimation methods in efficiency and accuracy. We also demonstrated that OIRT tracked students' ability growth dynamically and efficiently, it also predicted students' future performance with reasonable AUC given limited input features.

Keywords

Item Response Theory, Performance Factor Analysis, Online Learning, Bayesian Parameter Estimation, Variational Inference

1. INTRODUCTION

As time series data become increasingly prevalent in online learning system, tracking students' ability changes during their learning processes is important for the analysis of teaching and learning activity. There have been three commonly used models for estimating students' cognitive mastery: Item Response Theory (IRT) model is a general tool to provide a quantitative description of students' ability in academic testing. Knowledge Tracing (KT) model tries to predict a students' future performance through their historical interaction logs [5]. Performance Factor Analysis (PFA) [15] analyzes learning rates of students by considering multiple Knowledge Components (KCs) of each exercise item.

None of the above approaches are perfectly applicable to monitor students' ability changes in online learning. IRT roots on the assumption that students' true abilities are fixed [18], which may not be true in online learning environment, because student abilities are dynamic. Bayesian KT only estimates binary hidden states (either mastery or non-mastery) and models each KC separately. Standard IRT and PFA models are not able to perform real-time parameter estimation due to model format or estimation methods.

L. Peng and C. Wei. Online item response theory (OIRT) - tracking student abilities in online learning system. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 720–724, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license. <https://doi.org/10.5281/zenodo.6853004>

In this study, we propose an Online Item Response Theory Model (OIRT) to track students' ability changes in real-time fashion using both simulation and real data.

In summary, the contribution of the work is three-fold: (1) propose OIRT model by estimating students' initial abilities, item difficulties as well as ability changes for different KCs; (2) modify Variational Inference (VI) [24] under OIRT model to track students' ability changes; (3) compare the computational time and accuracy of the modified VI with other parameter estimation approaches, and demonstrate answer accuracy prediction by OIRT.

2. BACKGROUND

In this part, IRT and PFA models as well as common real-time parameter estimation approaches are briefly reviewed.

2.1 Item Response Theory Model

IRT is widely used in assessing student abilities and item difficulties due to its high interpretability. The one-parameter logistic (1PL) model [16] is given in Eq.1,

$$p(y_{ij}|\theta_i, b_j) = \frac{1}{1 + e^{-(\theta_i - b_j)}} \quad (1)$$

where y_{ij} is the i -th student's response to the j -th item. $y_{ij} = 1$ indicates a correct answer and 0 otherwise. θ_i denotes the ability of the i -th student and b_j denotes the difficulty of the j -th item. We developed our OIRT model based on 1PL model in Eq.1, but OIRT can be extended to 2 or 3PL IRT models [7,8] easily.

2.2 Performance Factor Analysis

IRT model only estimates a constant ability for each student and cannot model the changes of student abilities as learning proceeds [18]. To address this problem, especially in the adaptive online learning environment, Learning Factor Analysis (LFA) model [4] and PFA model [15] are proposed to further include the prior practice counts for each KC. Specifically, PFA model, an extension of LFA model, is given in Eq.2,

$$p(y_i = 1|\beta_k, \gamma_{i,k}, \rho_{i,k}) = \frac{1}{1 + e^{-\sum_{k=1}^K (-\beta_k + \gamma_{i,k} * s_{i,k} + \rho_{i,k} * f_{i,k})}} \quad (2)$$

Here, β_k is the difficulty of the k -th KC, $s_{i,k}$ and $f_{i,k}$ are the prior successes and failures of the i -th student on the k -th KC, $\gamma_{i,k}$ and $\rho_{i,k}$ are the learning rates of these observation counts, implying the effects of accumulated successes and failures ($s_{i,k}$ and $f_{i,k}$) on answer accuracy in the processes of learning.

Some other models also try to track the changes of student abilities in a short period [10, 13, 23]. The main principle here is

to estimate the ability change, $\Delta\theta_t$, based on students' responses to items. We also follow this principle by modeling $\Delta\theta_t$ using learning rate parameters and their corresponding practice counts, which will be introduced in Section 3.

3. ONLINE ITEM RESPONSE MODEL

Online Item Response Theory (OIRT) model is an extension of the existing PFA model. Suppose there are N students, M items covering a total of K KCs, the OIRT model is given in Eq.3,

$$p(y_{ij}|\theta_i, \bar{\gamma}_i^s, \bar{\gamma}_i^f, b_j) = \frac{1}{1 + e^{-(\theta_i - b_j + (\bar{\gamma}_i^s \odot \bar{s}_i + \bar{\gamma}_i^f \odot \bar{f}_i)^T \bar{T}_j)}} \quad (3)$$

where θ_i and b_j denote the i -th student's general ability and the j -th item's difficulty, respectively. Let K be the total number of KCs covered by all the items, \bar{s}_i and \bar{f}_i are $K * 1$ vectors containing successful and unsuccessful practice counts for the i -th student. $\bar{\gamma}_i^s$ and $\bar{\gamma}_i^f$ are the $K * 1$ learning rate vectors for \bar{s}_i and \bar{f}_i , respectively. \bar{T}_j is a pre-specified $K * 1$ distributional vector of KCs for item j . The \odot and \cdot are element-wise product and dot product, respectively.

OIRT contains four extensions compared to PFA model in Eq.2. (1) An initial ability θ_i for each student is added in OIRT due to the prior knowledge of students. (2) Note that modeling item difficulty as $\sum_k \beta_k$ in PFA is unreasonable in that the item with the same KCs will have same difficulty. To solve the problem, we added a unique difficulty b for each item in OIRT. (3) Instead of using a binary vector indicating which item covers which KC, we used a distributional vector \bar{T}_j to avoid a bias (working on items with more KCs will lead to higher ability gain when adding up the learning effects of all KCs covered by an item) towards the items with many KCs. To construct \bar{T}_j , suppose we have a total of $K = 3$ KCs, if item j covers KC 1 and 3, instead of representing the item-KC vector as $[1, 0, 1]$, we represent it as $\bar{T}_j = [1/2, 0, 1/2]$, whose sum is always equal to 1. (4) The parameters in OIRT will be updated in a real-time mode: once a student receive the feedback after answering an item, we update \bar{s}_i and \bar{f}_i and hence the corresponding learning rate vectors, this is a major difference between OIRT model and other IRT and PFA models, because of the dynamic updates of \bar{s}_i and \bar{f}_i , we can update the learning rate parameters, and hence track ability changes.

In online learning system, \bar{s}_i and \bar{f}_i are initialized to 0, which will then be accumulated once an item is completed by the student. Therefore, the general ability θ_i and item difficulty b_j will be estimated in the beginning, learning rates $\bar{\gamma}_i^s$ and $\bar{\gamma}_i^f$ will then be estimated as more practice data being collected.

4. PARAMETER INFERENCES OF OIRT

We applied and compared four parameter estimation methods in OIRT model: Maximum Likelihood Estimation (MLE) in Logistic Regression (LR), MCMC, EP and VI. We consider LR as a baseline and mainly introduce the other three methods under OIRT.

4.1 Markove Chain Monte Carlo

Markov Chain Monte Carlo (MCMC) [2, 3] can be directly used to perform real-time parameter estimation, because the prior of the interested parameter η at time t can be updated using the posterior based on the data at time $t - 1$, specifically, $p(\eta|Data_t) \propto p(Data_t|\eta) * p(\eta|Data_{t-1})$ given the conditional independence of data. It draws samples from the approximated posterior distributions from which the expectations and variances of the parameters are constructed. Researchers successfully applied MCMC to IRT parameter estimation [1, 14, 19, 20].

4.2 Expectation Propagation

Recall the parameters we need to estimate are $\eta = \{\bar{\theta}, \gamma^s, \gamma^f, \bar{b}\}$. Here γ^s and γ^f are $N * K$ matrices, $\bar{\theta}$ is an $N * 1$ ability vector and \bar{b} is an $M * 1$ item difficulty vector. We can reformulate the parameters as a long vector $\tau = [\bar{\gamma}_1^s, \bar{\gamma}_2^s, \dots, \bar{\gamma}_1^f, \bar{\gamma}_2^f, \dots, \bar{\theta}^T, \bar{b}^T]$. If the complete data is given, we can easily solve τ by a LR. However, data comes batch by batch, therefore, we can use Expectation Propagation (EP) [11, 12, 21].

Given N responses y_1, y_2, \dots, y_N , the posterior of η can be written as $p(\eta|y) \propto p(\eta) * p(y_1|\eta) * p(y_2|\eta) * \dots * p(y_N|\eta)$ if responses are conditionally independent. In EP, $p(y_i|\eta)$ is usually complicated function and approximated by \tilde{p}_i , $i \in 0, 1, 2, \dots, N$ (often chosen to be normal distribution). Here, $\tilde{p}_0 \approx p(\eta)$ and $\tilde{p}_i \approx p(y_i|\eta)$. Generally, we compute the following steps:

- (1) Initialize all \tilde{p}_i ,
- (2) Calculate the approximating posterior $q(\eta) = \frac{\prod_i \tilde{p}_i}{\int_{\theta} \prod_i \tilde{p}_i d\eta}$
- (3) Until all \tilde{p}_i 's converge for $i = 1, 2, 3 \dots N$:
 - i. Calculate cavity distribution $q^{(i)}(\eta) \approx \frac{q(\eta)}{\tilde{p}_i}$
 - ii. Update q by $argmin_{KL} KL(q(\eta) || q^{(i)}(\eta) * p(y_i|\eta))$
 - iii. Update $\tilde{p}_i \approx \frac{q(\eta)}{q^{(i)}(\eta)}$

In the KL divergence step for the IRT models, $q^{(i)}(\eta)$ is a normal density function but $p(y_i|\eta)$ is a logistic function, it is difficult to get a normal distribution approximation of this product. Therefore, some other approximation forms are proposed [6, 22] and we applied the approximation in [9] as well as its update rule in the KL step for logistic function, see [9] for details.

4.3 Variational Inference

Inspired from [24], we derived an ELBO function for our OIRT in Eq.4 by assuming the joint posterior distribution factors as $q(\eta|y) = q(\bar{\theta}|\bar{b}, y)q(\gamma^s|\bar{b}, y)q(\bar{b}|y)q(\gamma^f|\bar{b}, y)$,

$$ELBO = E_{q(\eta)}[\log p(y | \bar{\theta}, \bar{b}, \gamma^s, \gamma^f)] \quad (4)$$

$$-E_b[KL(q(\bar{\theta} | \bar{b}) || p(\bar{\theta} | \bar{b})) + KL(q(\gamma^s | \bar{b}) || p(\gamma^s | \bar{b})) + KL(q(\gamma^f | \bar{b}) || p(\gamma^f | \bar{b}))] - KL(q(\bar{b}) || p(\bar{b}))$$

For simplicity, we simplified Eq.4 as $ELBO = likelihood - KL_{\theta} - KL_b - KL_{\gamma^s} - KL_{\gamma^f}$. Then, the following algorithm is used to estimate parameters:

- (1) At time $t = 0$, initialize the priors of the parameters $p_0(\bar{\theta}), p_0(\bar{b}), p_0(\gamma^s), p_0(\gamma^f)$

- (2) Set shrink, enhance, decay hyperparameters.¹ Loop over iterations on loss optimization at each time t :
 - i. Update priors $p_t(\eta)$ at time t based on the combination of the approximated posterior $q_{t-1}(\eta)$ and the original prior $p_0(\eta)$ for each parameter in η : $p_t(\eta) = (1 - decay) * q_{t-1}(\eta) + decay * p_0(\eta)$
 - ii. Optimize $loss = likelihood - shrink^{iteration} * [(1 + enhance * \frac{g}{max}) * (KL_\theta + KL_b) + KL_y^S + KL_y^f]$ to obtain current posterior $q_t(\eta)$ to be used in each KL , g/max is the number of student-item pairs up to time t over the total number of pairs.

There are three differences compared to the standard VI: (1) We set the shrink factor to 0.95 after the first time point because the prior distributions now keep the information from the previous data and should not be shrunk. (2) We used a weighted average instead of directly replacing the prior at time t with the posterior at $t - 1$ so that the prior gets updated gradually and the previous information play in role smoothly. (3) We enhanced the KL_θ and KL_b gradually. At the first several sessions of student data, \vec{s}, \vec{f} are close to zero, therefore, student abilities and item difficulties are the only parameters being estimated in OIRT. As \vec{s}, \vec{f} increase, and since student abilities and learning rates are not identifiable (both are parameters of individual student), we gradually fixed student abilities and item difficulties so that the algorithm can focus on the estimation of those learning rates only. Our experience showed that shrink = 0.95, decay = [0.3, 0.5] and enhance = 7 is reasonable.

5. EXPERIMENTS AND RESULTS

We compared the performances of modified VI with MCMC, EP and LR in parameter estimation on two simulated datasets. We also demonstrated the online ability tracking of OIRT using a real data, and compared OIRT with XGBoost on answer prediction task on the real data. The software environment in these experiments is under Python 3.7, Pytorch-1.7.1, the hardware is Intel(R) Xeon(R) Gold 6130 CPU @ 2.10GHz, Tesla P4 GPU.

5.1 Simulation Studies

5.1.1 Standard Normal Distribution for Learning Rates

In the first experiment, we examined two conditions: 100 students and 500 students, both with 4 KCs and 100 items. The data simulations are as following:

- (1) We simulate student abilities, learning rates for s_i and f_i for each KCs, item difficulties from standard normal distributions independently. Generate and normalize the KC distribution vector for each item. Initialized \vec{s} and \vec{f} to be 0
- (2) Since the person-item pairs in each condition is 100*100 and 500*100, respectively, at each time point, we sample a random number of pairs from the remaining unused person-item pairs (in this case, person-item pairs could be generated sequentially)

- as the current data, and extract the corresponding parameters sampled in step (1) for each chosen person and item
- (3) Construct responses based on OIRT model in Eq.3
- (4) Update the \vec{s} and \vec{f} for each student at each session based on the responses in (3) and apply them in step (3) of next session
- (5) Repeat step (2) (3) (4) until all pairs are chosen

Table 1 shows the results for the 500 students condition. Under the standard normal distributions for the learning rates, LR (default setting in sklearn) has the highest accuracy in parameter estimation. MCMC is the second best, but it is more time-consuming. Even though the estimation accuracy of the modified VI is worse than MCMC and LR, its computational time is comparable to that of LR. EP has the worst parameter estimation performance due to its approximation issue discussed in Section 4.2. Similar results were obtained for the simulation with 100 students.

Table 1. Correlations with real values under standard normal parameter distribution: with 100 items and 4 KCs

Student s	Method s	ABI	DIFF	LS	LF	Time
500	LR	0.806	0.968	0.778	0.771	35.4s
	MCMC	0.656	0.977	0.702	0.725	5d
	EP	0.7	0.905	0.658	0.669	650m
	VI	0.706	0.789	0.532	0.491	84.5s

5.1.2 Non-standard Normal Distribution for Learning Rates

In the second experiment, student abilities and item difficulties were sampled independently from standard normal distribution, while learning rates for success and failure were sampled independently from non-standard normal distributions, $N(0.01, 0.03)$. Other simulation procedures remained the same.

In this case, the true distributions of the learning rates are no longer standard normal distributions, which may be more realistic because learning rates are usually small and positive. Since MCMC is time-consuming, we only compared VI, EP and LR. Results about the estimation accuracy with respect to abilities, difficulties and two learning rates are shown in Table 2.

It is clear from Table 2 that VI is still robust in estimating the learning rates when their true distributions are non-standard normal, it is also comparative to LR in ability and item difficulty estimations. VI is also more computationally efficient in dealing with more students and more KCs (500 items and 5KCs in Table 2). Similar results were obtained for 100 students.

Results about computational speed are shown in Figure 1 with varying students, KCs and item numbers. The computational time is the time each method spent on estimating all parameters throughout all generated sessions. The lines for EP and LR are incomplete because LR fails when it needs more than 256G memory and EP fails when it takes more than 5 days.

¹ Shrink controls the contribution of the KL terms in optimizing the loss function. Enhance gives more importance to KL terms as more data flows in, because the prior in KL at time t contains the information from the previous data that we want to keep. Decay controls the weight given to the posterior at time $t - 1$ in contributing to the prior at time t .

Table 2. Correlations with real values under non-standard Normal parameter distribution: with 500 items and 5 KCs

Students	Methods	ABI	DIFF	LS	LF	Time
500	LR	0.939	0.992	0.778	0.303	573s
	VI	0.936	0.969	0.658	0.661	145s
	EP	0.827	0.731	0.532	0.153	100h
	LR	0.939	0.992	0.778	0.303	573s

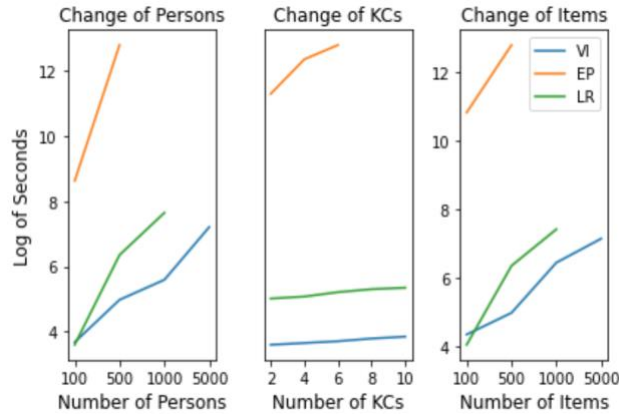


Figure 1. Computational time comparison

It is obvious to note that the results of the modified VI are better compared with that of the other methods in three aspects: (1) the computational speed of VI is faster as number of persons and items increase; (2) the modified VI gives better parameter estimation when the prior distributions disagree with the true distributions of the learning rates; (3) the modified VI supports real-time parameter estimations and requires less memories.

5.2 Real Data Study

In the third experiment, we used a real dataset, Riid public dataset² from Kaggle competition, to demonstrate the ability change tracking and answer prediction by OIRT.

We selected the event for a question being answered by the user (content_type_id=0) with prior question having explanation. We also removed the items and users with response frequencies fewer than 50. The data contained 6800 students, 1983 items and 146 KCs after preprocessing. We sorted the data by time the question was completed by the user, for ability tracking task, we used the whole data to estimate parameters; for answer prediction task, the first 90% was used train models, and the remaining 10% was used as testing set. We partition the data into 50 sessions with person-item pairs and feed one session into the model at a time.

We compared OIRT with XGBoost technique in answer prediction task. The reason for comparing with XGBoost is that both methods are single-layered and explainable models, which by nature are different and incomparable with the models based on deep neural networks. We only used ‘Timestamp’, ‘Tags’, ‘User ID’, and ‘Item ID’ as the input features for both OIRT and XGBoost. The ‘answered correctly’ was the label for the models. OIRT outperformed XGBoost in accuracy prediction of future question

responses with limited input features: AUC=0.702 vs 0.689, ACC=0.733 vs 0.717 (since the XGBoost in the competition uses complex feature engineering, its AUCs reported in the competition are much higher). OIRT also provides reasonable estimates for user ability and item difficulty due to its high correlation with the observed accuracy proportion for students and items (0.751, 0.696, respectively).

We randomly selected 2 users and plotted Figure 2 to show the ability change tracking of OIRT by comparing with the observed differences of the accuracy proportion between two adjacent time points, averaging all KCs at each session. The estimations are equal to $(\bar{\gamma}_i^s \odot \bar{s}_i + \bar{\gamma}_i^f \odot \bar{f}_i)^T \bar{T}_j$ in Eq.3 at each time t (below). The observed changes in accuracy proportion is equal to $(\# \text{correct KCs}_{1:t} - \# \text{correct KCs}_{1:t-1}) / \# \text{KC}_s$ for each user (above), indicating how many more accurate KCs completed by a user at time t relative to that at time $t - 1$.

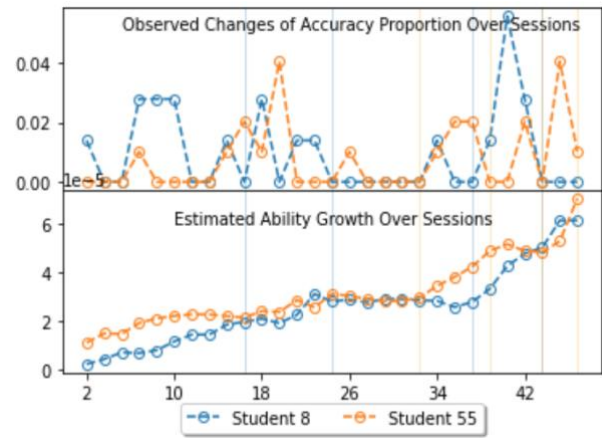


Figure 2. Students' ability tracking by OIRT

It can be seen when the observed increase in accuracy proportion are high between two adjacent time points, the estimated ability growth is more abrupt, such as sessions in the blue and orange windows for student 8 and student 55, respectively.

6. CONCLUSIONS

In this study, we developed OIRT model and modified VI parameter estimation method to track student abilities in real-time and predict answer correctness for online learning system. Results show that the modified VI can estimate the parameters fast and effectively despite of the difference between the priors and the true distribution of the learning rate parameters.

Although OIRT performs relatively well in different tasks introduced above, it takes the form of generalized linear model, which has parameter identification issue and limits its performance in the accuracy prediction for future questions. We only predict answer accuracy based on historical data for individuals, and didn't examine the prediction accuracy for new students, which will be explored more in future study.

² <https://www.kaggle.com/c/riid-test-answer-prediction/data>

7. REFERENCES

- [1] Albert, J. H. Bayesian estimation of normal ogive item response curves using Gibbs sampling. *Journal of educational statistics*, 17(3):251-269, 1992. <https://doi.org/10.2307/1165149>
- [2] Andrieu, C., N. De Freitas, Doucet, A., and Jordan, M. I. An introduction to mcmc for machine learning. *Machine learning*, 50(1):5-43, 2003.
- [3] Andrieu, C. and Thoms, J. A tutorial on adaptive mcmc. *Statistics and computing*, 18(4):343-373, 2008. <https://doi.org/10.1007/s11222-008-9110-y>
- [4] Cen, H., Koedinger, K., and Junker, B. Learning factors analysis—a general method for cognitive model evaluation and improvement. In *International Conference on Intelligent Tutoring Systems*, pages 164-175. Springer, 2006. https://doi.org/10.1007/11774303_17
- [5] Curi, M., Converse, G. A., Hajewski, J., and S. Oliveira. Interpretable variational autoencoders for cognitive models. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1-8. IEEE, 2019. DOI:[10.1109/IJCNN.2019.8852333](https://doi.org/10.1109/IJCNN.2019.8852333)
- [6] Hall, P., Johnstone, I., Ormerod, J., Wand, M., and Yu, J. Fast and accurate binary response mixed model analysis via expectation propagation. *Journal of the American Statistical Association*, 115(532):1902-1916, 2020. <https://doi.org/10.1080/01621459.2019.1665529>
- [7] Hambleton, R. K. and Cook, L. L. Latent trait models and their use in the analysis of educational test data. *Journal of educational measurement*, pages 75-96, 1977. <http://www.jstor.org/stable/1434009>.
- [8] Lord, F. A theory of test scores. *Psychometric monographs*, 1952. <https://psycnet.apa.org/record/1954-01886-001>
- [9] MacKay, D. J. The evidence framework applied to classification networks. *Neural computation*, 4(5):720-736, 1992. DOI: [10.1162/neco.1992.4.5.720](https://doi.org/10.1162/neco.1992.4.5.720)
- [10] Martin, A. D., and Quinn, K. M. Dynamic ideal point estimation via markov chain monte carlo for the us supreme court. *Political analysis*, 10(2):134-153, 2002. DOI: <https://doi.org/10.1093/pan/10.2.134>
- [11] Minka, T. Ep: A quick reference. *Techincal Report*, 2008.
- [12] Minka, T. P. Expectation propagation for approximate bayesian inference. *arXiv preprint arXiv:1301.2294*, 2013. <https://doi.org/10.48550/arXiv.1301.2294>
- [13] Park, J. Y., Cornillie, F., van der Maas, H. L., and Van Den Noortgate, W. A multidimensional irt approach for dynamically monitoring ability growth in computerized practice environments. *Frontiers in psychology*, 10:620, 2019. DOI: [10.3389/fpsyg.2019.00620](https://doi.org/10.3389/fpsyg.2019.00620)
- [14] Patz, R. J. and Junker, B. W. A straightforward approach to markov chain monte carlo methods for item response models. *Journal of educational and behavioral Statistics*, 24(2):146-178, 1999. <https://doi.org/10.2307/1165199>
- [15] Pavlik Jr, P. I., Cen, H., and Koedinger, K. R. Performance factors analysis—a new alternative to knowledge tracing. *Online Submission*, 2009.
- [16] Rasch, G. Studies in mathematical psychology: I. probabilistic models for some intelligence and attainment tests. 1960.
- [17] Settles, B., Brust, C., Gustafson, E., Hagiwara, M., and Madnani, N. Second language acquisition modeling. In *Proceedings of the NAACL-HLT Workshop on Innovative Use of NLP for Building Educational Applications (BEA)*. ACL, 2018. [10.18653/v1/W18-0506](https://doi.org/10.18653/v1/W18-0506)
- [18] Van der Linden, W. J. *Handbook of item response theory: Volume 1: Models*. CRC Press, 2016.
- [19] Van der Linden, W. J. and Jiang, B. A shadow-test approach to adaptive item calibration. *Psychometrika*, 85(2):301-321, 2020. doi: 10.1007/s11336-020-09703-8
- [20] Van der Linden, W. J. and Ren, H. A fast and simple algorithm for bayesian adaptive testing. *Journal of educational and behavioral statistics*, 45(1):58-85, 2020. <https://doi.org/10.3102/1076998619858970>
- [21] Wang, S. Expectation propagation algorithm, 2011.
- [22] Wang, S., Jiang, X., Wu, Y., Cui, L., Cheng, S., and Ohno-Machado, L. Expectation propagation logistic regression (explorer): distributed privacy-preserving online model learning. *Journal of biomedical informatics*, 46(3):480-496, 2013. doi: 10.1016/j.jbi.2013.03.008
- [23] Wang, X., Berger, J. O., and Burdick, D. S. Bayesian analysis of dynamic item response models in educational testing. *The Annals of Applied Statistics*, 7(1):126-153, 2013. <https://doi.org/10.48550/arXiv.1304.4441>
- [24] Wu, M., Davis, R. L., Domingue, B. W., Piech, C., and Goodman, N. Variational item response theory: Fast, accurate, and expressive. *arXiv preprint arXiv:2002.00276*, 2020. <https://doi.org/10.48550/arXiv.2002.00276>

Looking for the best data fusion model in Smart Learning Environments for detecting at risk university students

Wilson Chango¹, Rebeca Cerezo², Cristóbal Romero³

¹Pontifical Catholic University of Ecuador, Ecuador

²University of Oviedo, España

³University of Cordoba, España

wilson.chango@pucese.edu.ec, cerezorebeca@uniovi.es, cromero@uco.es

ABSTRACT

This paper proposes to discover which data fusion approach and classification algorithm produced the best results from smart classrooms data, and how useful would be the prediction models for detecting University students at risk of failing or dropout. The results showed that the best predictions were produced using ensembles and selecting the best attributes approach with discretized data; the REPTree algorithm demonstrated the highest prediction values. The best predictions also show the teacher what set of attributes and values are the most important for predicting student performance, such as the level of attention in theory classes, scores in Moodle quizzes and the level of activity in Moodle forums.

Keywords

Data fusion, student prediction models, smart classroom.

1. INTRODUCTION

Nowadays, new learning models are used in Higher Education such as Blended learning, Smart Learning and Multimodal learning.

Blended learning (b-learning) is an approach to learning and instruction that combines online educational materials and opportunities for interaction online with traditional place-based classroom methods, although the terms "blended learning", "hybrid learning", "technology-mediated instruction", "web-enhanced instruction", and "mixed-mode instruction" are often used interchangeably in research literature [5][6]. Its rise is due to the fact that there are some disadvantages to current e-learning environments such as inhibiting socialization resulting in a lack of face-to-face communication [7]. Otherwise, EDM has been widely used to improve and enhance learning quality, as well as in the pursuit of pure research objectives, which tend to improve our understanding of the learning process [8]. In this context, it is still a challenge to predict student learning achievement in blended learning environments combining online and offline learning [1, 6], making data fusion techniques necessary.

Smart learning environments (SLEs) have been recently defined [9] as learning ecologies wherein students perform learning tasks

C. Romero, W. Chango, and R. Cerezo. Looking for the best data fusion model in smart learning environments for detecting at risk university students. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 725–728, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853089>

and/or teachers define them with the support provided by tools and technology. SLEs can encompass physical or virtual spaces in which a system senses the learning context and process by collecting data, analyzes the data, and consequently reacts with customized interventions that aim at improving learning [9].

In Multimodal Learning Analytics (MLA), learning traces are extracted not only from log-files but also from digital documents, recorded video and audio, pen strokes, position tracking devices, biosensors, and any other data source that could be useful for understanding or measuring the learning process. One important question in MLA is how to combine, or fuse, the data extracted from different modalities in order to provide a more comprehensive view of learners' outer and inner processes [1].

In this study we propose applying different data fusion approaches and classification algorithms to data gathered from several sources (theory classes, practical sessions, online sessions, and final exams) in a blended, smart, multimodal course in order to predict the students' final academic performance [11]. Data fusion, or information fusion, is the study of efficient methods for automatically or semi-automatically transforming information from different sources and different timepoints into a representation that provides effective support for human or automated decision making. Specifically, data fusion can reduce the size and dimensions of data, optimize the amount of data and extract useful information [10]. There are different types of multimodal fusion approaches such as: feature-level or early fusion, decision-level or later fusion and Hybrid fusion.

2. EXPERIMENTS

We used information from 57 first-year electrical engineering students at the University of Cordoba (Spain) in the Introduction to Computer Science course during the first semester of academic year 2017-2018. We have gathered all the information from four data sources: theory classes, practical classes, on-line sessions and final exam. The first three data sources gave us the input attributes and the final exam, the output attribute or class to predict. The students all gave their written consent to being recorded, after being informed about the study, and to have their data from practical and online sessions in Moodle collected for the study.

We have used four different data fusion approaches (merging all attributes; selecting the best attributes; using ensembles; and using ensembles and selecting the best attributes) and several white-box classification algorithms with the datasets. Then, we compare the predictions produced by the models (%Accuracy and ROC Area) to discover the best approach and classification model so that it is used for predicting students' final performance.

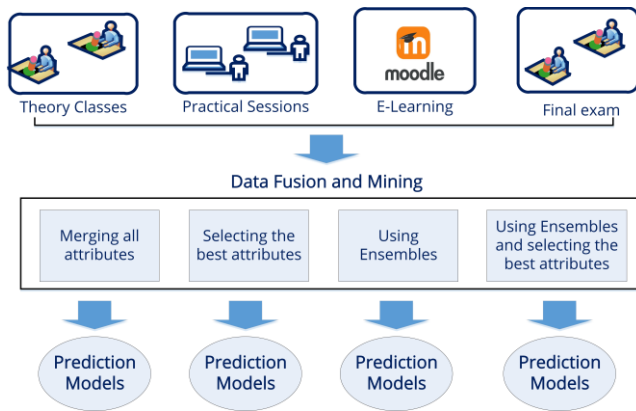


Figure 1. Proposed data fusion and mining methodology for predicting students' performance from multiple data sources.

2.1 Experiment 1: Merging all attributes

In experiment 1 we applied the classification algorithms to a single file with all the attributes merged. Firstly, we fused the different values of the 6 attributes collected in the theory and practical sessions in order to have just one single value for each attribute. In our case, we had 15 values (15 lectures) for each one of the 4 attributes collected in the face-to-face theory classes and 10 (10 sessions) and 5 (5 practicals) values respectively for each of the 2 attributes for face-to-face practice sessions. Fusing the 4 values about the on-line sessions was not necessary because the specific tool that we used for preprocessing the Moodle logs gave a single value for each attribute directly.

Table 1. Results produced by merging all attributes

	NUMERICAL DATA		DISCRETIZED DATA	
	% Accuracy	AUC	%Accuracy	AUC
Jrip	77.1930	0.8440	78.9474	0.8880
Nnge	80.4561	0.8760	75.4386	0.8630
PART	78.9474	0.8640	80.4561	0.9170
J48	75.4386	0.8640	78.9474	0.8780
REPTree	75.4386	0.8630	76.6667	0.8480
Randomtree	70.1754	0.7820	73.6842	0.8180
Avg.	76.2749	0.8488	77.3567	0.8687

Table 1 shows that the best results (highest values) were produced by Nnge (80.45 %Acc) and Part (80.45%Acc and 0.91 AUC) algorithms. On average, most of the algorithms exhibited slightly improved performance in both measures when using discretized data.

Table 2. PART decision list when merging all attributes

IF Moodle.Quiz = Medium AND Theory.Attention = Medium THEN Pass
IF Moodle.Quiz = High THEN Pass
IF Theory.Attention = Low AND Moodle.Forum = Low THEN Dropout
IF Moodle.Quiz = Low THEN Fail
ELSE Pass
Number of Rules : 5

This prediction model (see **Table 2**) consists of 5 rules that show that the students who pass the course are students who have medium scores in Moodle quizzes and also pay attention in theory classes, or students who simply have high scores in Moodle

quizzes. The students who drop out from the course are students who pay little attention in theory classes and also show low activity in the Moodle forum. The students who fail the course are the students who get low scores in the Moodle quizzes. The remaining students are classified as passing.

2.2 Experiment 2: Selecting the best attributes

The selection of characteristics is important in the classification process by reducing not only the dimensions of the characteristic set but also the additional calculation time required for the classification algorithms. We used the well-known CfsSubsetEval (Correlation-based Featured Selection) method [11] provided by the WEKA tool [36]. It evaluates the worth of a subset of attributes by considering the individual predictive ability of each feature along with the degree of redundancy between them. Starting from our initial 10 input attributes, we produced two sets of 3 different optimal attributes for the numerical and discretized datasets (see Table 3).

Table 3. Results obtained when selecting the best attributes

	NUMERICAL DATA		DISCRETIZED DATA	
	% Accuracy	AUC	%Accuracy	AUC
Jrip	80.7018	0.8490	82.4561	0.9140
Nnge	82.4561	0.9140	78.9474	0.8430
PART	77.1930	0.8750	80.7018	0.9140
J48	80.7018	0.8680	82.4561	0.9230
REPTree	77.1930	0.8940	78.9474	0.8880
Randomtree	75.4386	0.8320	82.4561	0.9170
Avg.	78.9474	0.8720	80.9942	0.8998

Table 3 shows that the best results (highest values) were produced by Jrip (82.45%Acc), Nnge (80.45 %Acc), and J48 (82.45 %Acc and 0.92 AUC) algorithms. Again, on average most of the algorithms exhibited slightly improved performance in both measures when using discretized data.

Table 4. J48 pruned tree when selecting the best attributes

IF Moodle.Quiz = Low
Moodle.Forum = Low
Theory.Attention = Low THEN Dropout
Theory.Attention = Medium THEN Fail
Theory.Attention = High THEN Fail
Moodle.Forum = Medium THEN Fail
Moodle.Forum = High THEN Fail
ELSE IF Moodle.Quiz = Medium
Theory.Attention = Low THEN Fail
Theory.Attention = Medium THEN Pass
Theory.Attention = High THEN Pass
ELSE IF Moodle.Quiz = High THEN Pass
Number of Leaves: 9
Size of the tree: 13

This prediction model (see Table 4) is a decision tree with 9 leaves that can be transformed into 9 prediction rules. These rules show that the students who pass the course are students who have medium scores in Moodle quizzes and also pay medium to high attention in theory classes, or students who simply have high scores in Moodle quizzes. The students who drop out from the course are students who have low scores in Moodle quizzes, show low activity in the Moodle forum, and also pay little attention in theory classes. In addition, students who fail are the students that

have low scores in Moodle quizzes, show low activity in the Moodle forum and pay medium to high attention in theory classes. There are also other failing student profiles: students who have medium scores in Moodle quizzes and also pay little attention in theory classes; students who have low scores in Moodle quizzes, show low activity in the Moodle forum, and pay medium to high attention in theory classes.

2.3 Experiment 3: Using ensembles

In experiment 3 we applied an ensemble of classification algorithms to each different source of data. However, instead of merging all of the attributes from the 4 data sources into a single file, we added the students' final academic performance to each dataset. This produced three sets of datasets (6 files in total): two files (numerical and discrete version) for the theory classes with 4 input attributes and 1 output attribute or class; two files (numerical and discrete version) for the practical session with 2 input attributes and 1 output attribute or class; and two files (numerical and discrete version) for the online Moodle sessions with 4 input attributes and 1 output attribute or class.

A classifier is accurate if it works better than a random prediction of the data; sets perform better when base models are unstable with output undergoing significant changes in response to small changes in training data. We used the well-known Vote [4] automatic combining machine learning algorithm provided by WEKA. It produces better results than individual classification models, if the classifiers of the sets are accurate and diverse. Vote adaptively resamples and combines so that resampling weights are increased for those cases more often misclassified and the combination is done by weighted vote.

Table 5. Results obtained when using ensembles.

	NUMERICAL DATA		DISCRETIZED DATA	
	% Accuracy	AUC	%Accuracy	AUC
Jrip	82.4561	0.9230	85.9649	0.9380
Nnge	77.1930	0.8770	77.1930	0.8770
PART	80.7018	0.9040	82.4561	0.9130
J48	82.4561	0.9110	82.4561	0.9220
REPTree	82.4561	0.9230	82.4561	0.9220
Randomtree	77.1930	0.8360	79.9474	0.9170
Avg.	80.4094	0.8957	81.7456	0.9185

Table 5 shows that the best results (highest values) were produced by Jrip (85.96 %Acc and 0.93 AUC). Once again, on average most of the algorithms exhibited slightly improved performance in both measures when using discretized data

Table 6. JRIP when using ensembles.

JRIP rules (Theory):

```

=====
IF (Theory.Attendance = High) THEN Pass
IF (Theory.Attention = Low) THEN Dropout
ELSE Dropout
Number of Rules : 3

```

JRIP rules (Practice):

```

=====
IF (Practice.Attendance = High) and (Practice.Score = High)
THEN Pass
IF (Practice.Attendance = Low) and (Practice.Score = Low)
THEN Fail
ELSE Dropout
Number of Rules : 3

```

JRIP rules (Moodle):

```

=====
IF (Moodle.Task = Low) and (Moodle.Quiz = Low) THEN Fail
IF (Moodle.Quiz = Medium) and (Moodle.Forum = Low) THEN
Fail
IF (Moodle.Task = Medium) THEN Pass
IF (Moodle.Quiz = High) THEN Pass
ELSE Dropout
Number of Rules : 5

```

This prediction model (see Table 6) the students who regularly attend theory classes pass the course; the students who exhibit low attendance finally drop out. The students who regularly attend practical classes and exhibit high performance in those practical classes then pass the entire course. In contrast, the students who rarely attend practical classes and have low performance in practicals then fail the entire course. The students who upload a moderate number of activities to the Moodle platform or get high scores in Moodle quizzes are students who pass the course; and logically, the students who upload a low number of activities to the Moodle platform and get low scores in Moodle quizzes are students who fail the course, but the students with medium performance in quizzes and low contributions to the forum also fail.

2.4 Experiment 4: Using ensembles and selecting the best attributes

In experiment 4 we applied an ensemble of classification algorithms to the best attributes from each different source of data. Firstly, we selected the best attributes for each of the three different sets of datasets (6 files in total) generated in experiment 3. For that, we again used the well-known CfsSubsetEval attribute selection algorithm. The best result with our data was obtained when combining a weight of 1 for Theory and Practical with a weight of 2 for Moodle by using the average as combination rule for weights. We executed the six classification algorithms as base or individual classification models of our Voting method for the 6 previously generated summary datasets (see Table 7).

Table 7. Results obtained when using ensembles and selection of the best attributes.

	NUMERICAL DATA		DISCRETIZED DATA	
	% Accuracy	AUC	%Accuracy	AUC
Jrip	82.4561	0.9170	84.2105	0.9310
Nnge	80.7018	0.9020	78.9474	0.8900
PART	80.7018	0.9010	82.4561	0.9350
J48	82.4561	0.8990	84.2105	0.9350
REPTree	84.2105	0.9130	87.4737	0.9420
Randomtree	77.1930	0.9160	82.4561	0.9330
Avg.	81.2866	0.9080	83.2924	0.9277

Table 7 shows that the best results (highest values) were produced by REPTree (87.47 %Acc and 0.94 AUC). Again, on average, most of the algorithms exhibited slightly improved performance in both measures when using discretized data.

Table 8. RepTree when using ensembles with selecting the best attributes.

REPTree (Theory)

```

=====
IF Theory.Attention = Low THEN Dropout
IF Theory.Attention = Medium THEN Fail
IF Theory.Attention = High THEN Fail

```

Size of the tree : 4
REPTree (Practice)

IF Practice.Attendance = Low THEN Dropout
IF Practice.Attendance = Medium THEN Fail
IF Practice.Attendance = High
| AND Practice.Score = Low THEN Fail
| OR Practice.Score = Medium THEN Fail
| OR Practice.Score = High THEN Pass

Size of the tree : 7
REPTree (Moodle)

IF Moodle.Quiz = Low
| AND Moodle.Forum = Low THEN Dropout
| OR Moodle.Forum = Medium THEN Fail
| OR Moodle.Forum = High THEN Fail
ELSE IF Moodle.Quiz = Medium THEN Pass
ELSE IF Moodle.Quiz = High THEN Pass
Size of the tree : 7

This prediction model (see Table 8) is also a combination of three models that show differential student behavior related to theory, at the same time, students exhibiting medium or high attention, or medium to high Moodle forum participation fail; those demonstrating medium practical attendance or high practical attendance plus low or medium practice score also fail. The students that demonstrate high practical attendance and performance pass, as do the students with medium to high scores in Moodle quizzes.

In general, we can see that these white-box models are very useful for explaining to the teacher how the predictions of pass, fail or dropout are arrived at. The teacher can discover what the main predictive attributes and values are directly from the background of the IF-THEN rules.

3. CONCLUSIONS

This paper proposes to use four different data fusion approaches and six white-box classification algorithms to predict university students' academic performance, from multiple-source and multimodal data in smart learning environments. We carried out 4 experiments to answer two research questions as conclusion:

- Which data fusion approach and classification algorithm produce the best results from our data? The REPTree classification algorithm produced the best results in this approach from discretized summary data.
- How useful are the prediction models we produce to help teachers detect students at risk of failing courses or dropping out? The white-box models we produced give teachers very understandable explanations (IF-THEN rules) of how they classified the students' final performance or classification.

In the future, we intend to carry out more experiments in order to improve our process and to overcome some limitations:

- Analyzing the video automatically rather than manually or semi-automatically. Processing the video recordings automatically would gather information more efficiently compared to manual coding with the multiple modalities that characterize the classroom[1].
- Using raw data and other specific data fusion techniques. We used a basic Naïve and knowledge-based fusion method that uses summary data. However, there are many mathematical theories for fusing data [2] such as

Probability-based methods (PBM) and Evidence reasoning methods (EBM) that we can use with raw data.

- Using more sources of information, including videos of practicals and on-line session interaction with Moodle; audio from theory classes and practicals, text analytics or text mining of what students write during theory classes, practicals or in Moodle.

4. ACKNOWLEDGMENTS

This work would not have been possible without the funding from the Ministry of Sciences and Innovation I+D+I PID2020-115832GB-I00 and PID2019-107201GB-I00.

5. REFERENCES

- [1] Chan, M.C.E., Ochoa, X. and Clarke, D. 2020. Multimodal Learning Analytics in a Laboratory Classroom. *Machine Learning Paradigms*. M. Virvou, E. Alepis, G.A. Tsihrintzis, and L.C. Jain, eds. Springer International Publishing, 131–156.
- [2] Ding, W., Jing, X., Yan, Z. and Yang, L.T. 2019. A survey on data fusion in internet of things: Towards secure and privacy-preserving fusion. *Information Fusion*. 51, (Nov. 2019), 129–144. DOI:<https://doi.org/10.1016/j.inffus.2018.12.001>.
- [3] Jo, I.-H., Park, Y., Kim, J. and Song, J. Analysis of Online Behavior and Prediction of Learning Performance in Blended Learning Environments. 18.
- [4] Kuncheva, L.I. 2014. *Combining pattern classifiers: methods and algorithms*. Wiley.
- [5] Maarop, A.H. and Embi, M.A. 2016. Implementation of Blended Learning in Higher Learning Institutions: A Review of Literature. *International Education Studies*. 9, 3 (Feb. 2016), 41. DOI:<https://doi.org/10.5539/ies.v9n3p41>.
- [6] Martyn, M. The Hybrid Online Model: Good Practice. *2003*. 26(1), 18–23.
- [7] Moskal, P., Dziuban, C. and Hartman, J. 2013. Blended learning: A dangerous idea? *The Internet and Higher Education*. 18, (Jul. 2013), 15–23. DOI:<https://doi.org/10.1016/j.iheduc.2012.12.001>.
- [8] Romero, C. and Ventura, S. 2020. Educational data mining and learning analytics: An updated survey. *WIREs Data Mining and Knowledge Discovery*. 10, 3 (May 2020). DOI:<https://doi.org/10.1002/widm.1355>.
- [9] Tabuenca, B., Serrano-Iglesias, S., Carruana-Martin, A., Villa-Torrano, C., Dimitriadis, Y. A., Asensio-Perez, J. I., & Kloos, C. D. 2021. Affordances and core functions of smart learning environments: A systematic literature review. *IEEE Transactions on Learning Technologies*. 14(2), 129–145.
- [10] Worsley, M. 2014. Multimodal Learning Analytics as a Tool for Bridging Learning Theory and Complex Learning Behaviors. *Proceedings of the 2014 ACM workshop on Multimodal Learning Analytics Workshop and Grand Challenge* (Istanbul Turkey, Nov. 2014), 1–4.
- [11] Z. Zacharis, N. 2016. Predicting Student Academic Performance in Blended Learning Using Artificial Neural Networks. *International Journal of Artificial Intelligence & Applications*. 7, 5 (Sep. 2016), 17–29. DOI:<https://doi.org/10.5121/ijaia.2016.7502>.

Distance measure between instructor-recommended and learner's learning pathways

Marie-Luce Bourguet
Queen Mary University of London
marie-luce.bourguet@qmul.ac.uk

Yushan Li
Beijing University of Posts and Telecommunications
liyushan@bupt.edu.cn

ABSTRACT

We propose a distance measure between instructor-recommended and learners' learning pathways and discuss how it may be used. The context of this study is a 15-week undergraduate course. The digital learning activities consisted in reading materials, interactive videos, quizzes and worksheets. They allowed the collection of many time-stamped digital traces. The course instructor recommended to engage in the activities regularly and prior to attending weekly synchronous tutorials. However, learning was largely self-regulated. This led to different learning strategies, which were captured and visualised in the form of learning pathways. Several patterns emerged: "regular learners" followed the recommendations; "irregular/selective learners" tended to engage in some activities only and irrespectively of the timing of the tutorials; and "late learners" waited until the end of the course to start engaging. We have formulated a distance measure (D) between the students' and the instructor-recommended learning pathways and found D to be a good indicator of a learner's time management and self-regulation skills. D was also found to have a moderate negative correlation ($r = -0.6087$) with exam marks, implying that the visualisation of learning pathways combined with a distance measure could be useful to raise students' awareness of their lack of self-regulation skills.

Keywords

Self-regulated learning, digital traces, learning pathways, distance measure.

1. INTRODUCTION

In Self-Directed Learning (SDL), the responsibility to learn shifts from an external source (typically a teacher) to the individual learner. In its broadest meaning, SDL describes a process by which individuals take the initiative in identifying their learning needs, formulating goals, identifying resources for learning, choosing and implementing appropriate strategies, and evaluating learning outcomes [15]. Many consider self-direction to be a fundamental difference between children and adults in a learning situation; and many argue that higher education is where and when learners should be given the tools to become independent self-directed learners, and educators should move away from an authoritative position to take a role of facilitator of learning.

The goal of SDL is not necessarily to become a fully autonomous learner. In a formal learning situation such as a university course, it

should be seen as a collaborative process between the instructor and the learner [11]. SDL is multidimensional and can be perceived through the notions of control [7] and personal responsibility [6] along a continuous scale. When the learning tasks are generated by the instructor, it may be more appropriate to talk about self-regulation [24], a notion that is close to self-direction and encompassed by it [16][18]. Self-Regulated Learning (SRL) strategies include rehearsal, elaboration, organisation, critical thinking, time management, effort regulation, peer learning and seeking help [4].

The design of learning materials and activities in a Learning Management System (LMS) can affect learners' self-regulation strategies [12]. There is convincing evidence that people who are given some initiative in their learning learn more and learn better than people who are passively being taught [10][14]. In online learning settings characterised with autonomy, such as Massively Open Online Courses (MOOCs), self-regulation becomes a critical factor for success [1][2][8]. When comparing online and blended learner's SRL strategies, Broadbent [5] has found that they differed in their use, but that overall predictors of grade were equivalent: time management, elaboration and rehearsal strategies were found to be key predictors. There seems to be general agreement among researchers that effective time management especially is an essential element of SRL [22].

Time management can be short-range and focus on the day-to-day planning and organisation of time or be long-range and used to regulate effort [3][17]. Studies exploring the academic procrastination (i.e., the intentional delay of an intended course of action) of undergraduate students have highlighted the relationships between procrastination tendencies and poor self-efficacy for self-regulation and low academic achievements, e.g., [19] [13]. In [9], procrastination is described as "self-regulation failure of performance". Negative relationships between procrastination, self-control and self-esteem [21], as well as health, wealth and happiness [23] have also been demonstrated.

Poor SRL and time management (such as procrastination) typically result in a learning pathway that is significantly different from the optimal or instructor-recommended pathway [20], where a *learning pathway* is defined as the route taken by a learner through a range of learning activities, which allows them to build knowledge progressively. In this paper, we present a study to understand if students' chosen learning pathways can be indicators of poor SRL skills and be predictors of course performance when they differ significantly from the instructor-recommended pathway. With this objective in mind, we have formulated a distance measure (D) between the students-generated and the instructor-recommended learning pathways.

M.-L. Bourguet and Y. Li. Distance measure between instructor-recommended and learner's learning pathways. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 729–733, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853087>

2. METHOD

2.1 Participants and Context

The context of the study is a 15-week course on Multimedia Fundamentals, delivered online during the Autumn 2021 to a rather large class of 234 3rd year undergraduate students on a Telecommunication Engineering programme. The learning objectives for this course are: (1) to be able to discuss and compare multimedia data representations; (2) to relate data properties, representation, and size; (3) to apply digital signal processing techniques for the compression of multimedia data; and (4) to describe modern multimedia standards and formats. It covers the following nine topics: digitisation, colour models, image, sound and video representations, lossless compression techniques, JPEG, MPEG, perceptual encoding (MP3), and digital broadcasting.

The students were all Chinese nationals, between 20- and 21-year-old, enrolled on a 4-year UK-China transnational programme delivered in China. About 40% of the students are female. For the duration of the course, the students were physically on campus in Beijing, while the instructor was in another country.

The digital learning activities made available on the LMS consisted in reading materials, short interactive videos (H5P technology), quizzes and exercise worksheets. In each interactive video, various exercises were embedded, which encouraged active learning and served as formative assessment: multiple choice questions, drag and drop exercises, fill in the blank exercises, select the right words exercises, etc. For each topic, further formative assessment was provided in the form of a 5-question quiz. The questions were randomly taken from a question bank, and each attempt could retrieve a different set of questions from the bank. Students were encouraged to attempt the quizzes several times to be exposed to a greater variety of questions. In addition, exercise worksheets could be downloaded, completed, and submitted again to the LMS. These worksheets could not be automatically marked, so no marks were allocated to them, but marks gained from the video embedded exercises and the quizzes were visible to the students and used for formative assessment only. At the end of the course, students would get 2 marks for completing at least 66% of the online learning activities (which included accessing the reading materials, interacting with the videos, answering the quizzes and uploading the completed worksheets), 1 mark for completing between 33% and 65% of these activities, and 0 mark if 32% or less had been completed. A total of 65 digital learning activities were proposed for which time-stamped digital traces could be collected, as summarised in Table 1.

In addition, weekly online synchronous tutorials were scheduled to answer students' questions, give live demonstrations, explain some of the topics in more detail, and discuss the quiz questions and worksheet exercises. Each week, an open-ended question was posted in a discussion forum for students to debate between themselves. Two mandatory labs were also scheduled, in week 4 and 10 respectively, for which reports had to be submitted as part of the summative assessment. In the first lab, students used Matlab to create audio spectrograms and observe various sound signals; in the second lab, they programmed a simple JPEG encoder and decoder. By the end of the course, the summative assessment was made of 80% final closed-book examination; 13% lab reports; 5% in-class test (a 20-question quiz generated from the same question bank as the weekly quizzes); and 2% for completion of the online activities.

The learning activities (reading material, interactive videos, quiz and worksheet) were released on a weekly basis, the week before the corresponding tutorial. For example, the introduction material

was released in week 0 and covered in the week 1 tutorial; the digitisation topic activities were released in week 1 and covered in the week 2 tutorial, etc. An activity released on the LMS would then remain available until the end of the course and could be attempted as many times as desired, including during the revision period (weeks 12 to 14) and until the final examination, scheduled in week 15.

The lecturer recommended to engage in learning activities weekly, prior to attending the tutorials. However, the learning was mostly self-regulated as students could choose when to engage in the learning activities, or even not to engage at all. Two hours a week was the recommended time to engage in the learning activities, and time and space were scheduled in the students' timetable for self-studying, but attendance at these sessions was not mandatory nor recorded. Most students chose to engage at different times and in their private space. The responsibility given to the students for their learning led to various strategies, which, thanks to the digital traces could be captured in the form of various learning pathways.

Table 1. Number of digital learning activities with digital traces per type (reading material, video, quiz, worksheet) and per course topic: 65 activities in total.

Topic (week)	Reading	Videos	Quiz	Worksheet	TOTAL
Introduction (w1)	2	0	0	0	2
Digitisation (w2)	1	3	1	1	6
Colour (w3)	1	4	1	1	7
Images (w4)	1	3	1	1	6
Video/Sound (w5)	1	5	1	1	8
Compression (w7)	0	4	1	1	6
JPEG (w8)	1	5	1	1	8
MPEG (w9)	1	6	1	1	9
MP3 (w10)	1	7	1	1	10
Broadcasting (w11)	1	0	1	1	3
TOTAL	10	37	9	9	65

2.2 Data Collection and Analysis

Each week and for each student the following data were collected: (1) date of latest engagement or attempt in each available learning activity on the LMS; and (2) updated video and quiz marks (students could improve their marks by re-engaging in the activities). The data were then anonymised: students names and email addresses were removed from the records, only student enrollment numbers were preserved to be able to track a student's activity across the duration of the course.

Data collection was constrained by the fact that, at any point in time, only one time-stamped digital trace would remain on the LMS for each activity: a new attempt at the same activity would overwrite the previous trace. It was thus important to carefully plan the timing of the data collection and to collect data repeatedly and regularly. We collected data the day before each weekly tutorial, when the material related to that tutorial had already been available on the LMS for a week. That way, we could capture whether a student had engaged in the learning activities before (as recommended) or after the corresponding tutorial. Data were collected weekly until the date of the final examination in week 15.

3. RESULTS

3.1 Learning Pathways

Figure 1 shows several examples of learning pathways. On the horizontal axis, the 65 learning activities are plotted in chronological order. The vertical axis shows all the teaching weeks of the semester. The gaps between weeks 3 and 4 and between weeks 5 and 6 correspond to weeks with no scheduled tutorial. The graphs indicate when a learning activity has been completed for the first time. The recommended pathway represents the lecturer’s intent, where each learning activity has been completed by the time of the corresponding tutorial.

Several patterns emerged as illustrated in the examples of Figure 1. Regular learners are students who followed the lecturer’s recommendations. Their pathway is typically just below the recommended pathway, showing that they have engaged in the learning activities within the week preceding the tutorial. Irregular or selective learners tended to choose the type of activities they engaged in (e.g., some students used only the videos, others only the quizzes) and often did this irregularly and irrespectively of the timing of the tutorials. Finally, the late learners are students who seldom engaged during the semester and typically used the learning activities at the end of the course to prepare for the final examination (i.e., combining massing and procrastination).

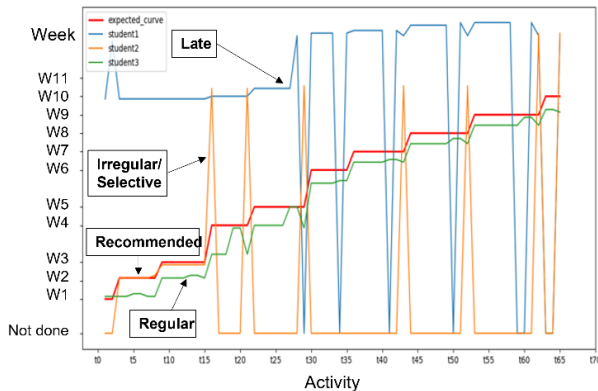


Figure 1. Recommended pathway and examples of regular, irregular/selective, and late learners’ pathways.

3.2 Distance Measure

We have formulated a distance measure (D) between a student’s and the lecturer-recommended learning pathways. It is calculated by adding together all the differences (in number of days) between when an activity was due and when it has been completed for the first time (see equation below). When an activity has not been completed, the maximum distance between the expected day and the end of the course (date of the examination in week 15), plus one day, is added to the sum. The sum is then normalised, dividing it by the maximum distance value (corresponding to a student who has not completed any of the activities) and multiplying by 100. D ranges from 2.38 (minimum distance found) to 96.9 (maximum distance found). The mean value of D is 31.61 (std = 25.95). Using K-means clustering (k = 3), the pathways were then grouped into three clusters.

$$Distance = \sum_{k=1}^{65} |expected_{day} - completed_{day}|$$

$$D = \frac{Distance}{Max\ Distance} \times 100$$

The first (low distance) cluster contains 47 pathways (20% of the students). These students can be classified as regular learners (see Figure 2). The maximum value of D in this cluster is 25.5, which can be used as a threshold value to “qualify” as a regular learner. All the pathway examples of Figure 2 follow the recommended pathway fairly closely, but each present a few “glitches”. The pathway with distance D = 8.30 shows that one of the latest activities has been missed, while another activity has been a bit delayed: until week 11 (final topic of the course) this student had been a very regular learner. The pathway with distance D = 12.81 shows that one of the early activities has been completed with a significant delay, while a few other activities were slightly delayed. It is possible that this student joined the course a bit late, did not understand what was demanded of them at first, or that it took them a bit of time to adopt the correct strategy (some “catch up” seems to have been done around week 9). The pathway with distance D = 17.37 is characterised by three undone activities, and one significantly delayed activity. The distance measure in this cluster seems to be a good indicator of the number and gravity of the “glitches”.

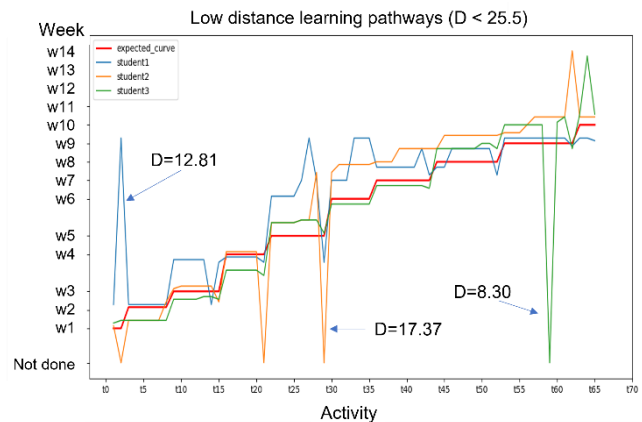


Figure 2. Examples of low distance (D=8.30; D=12.81; D=17.37) learning pathways (including the recommended pathway).

The second (middle distance) and largest cluster contains 128 pathways (55% of the students). The value of D in this cluster ranges between 25.5 and 56.8 (see examples in Figure 3). The examples of Figure 3 show that the students started as regular learners, but they shifted to selective or late learner profiles after some time (around week 5). Although the three values of D are close (28.61, 31.64 and 31.96 respectively), the pathways look quite different. The distance measure does not allow us to precisely discriminate between the selective and the late learners, so more work is needed in this respect. However, a student in this cluster may be described as an “irregular learner”.

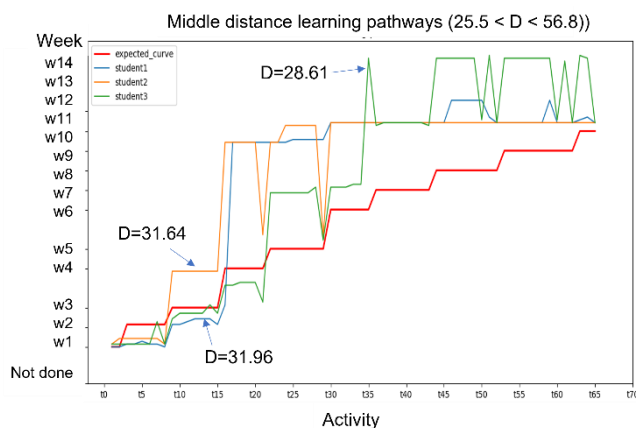


Figure 3. Examples of middle distance ($D=28.61$; $D=31.64$; $D=31.96$) learning pathways (for reference, the recommended pathway is also shown).

Finally, the third (high distance) cluster contains 59 pathways (25% of the students) (see Figure 4). In this cluster, the students adopted either a selective or a late learning strategy from the start of the course. Some of these students concentrated on one type of activity only, others did all the activities, but very late in the semester, i.e., when they started revising for the final examination. The pathway with distance $D = 60.30$ is characteristic of a late learner: all the activities have been completed but were all seriously delayed. The pathway with the highest distance ($D = 96.87$) is characteristic of a learner who almost never engaged in any of the activities. The pathway with a slightly lower distance ($D = 91.22$) shows a learner who is both selective and late. A high distance measure seems to be a good indicator of poor time management and lack of engagement.

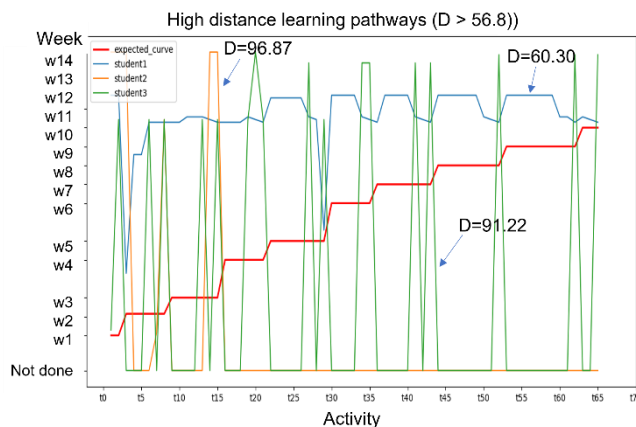


Figure 4. Examples of high distance ($D=60.30$; $D=91.22$; $D=96.87$) learning pathways (for reference, the recommended pathway is also shown).

Finally, we found D to have a moderate but significant negative correlation ($r = -0.6087$) with the students' final examination marks.

4. DISCUSSION

The negative correlation ($r = -0.6087$) between exam performance and the distance to the recommended learning pathway confirms that regular learning is conducive to deeper learning. There could be several reasons for this: (1) having studied the learning material before attending a tutorial allows students to make the best of the

time they have with the lecturer as they can prepare questions and better understand the additional content provided in the tutorial; (2) students have more time to assimilate and reflect on their learning before sitting the final examination; (3) it shows better time management skills, which are likely to have been effective also when preparing for the final examination.

As future work, we plan to test if, at the level of each student, the visualisation of their learning pathway combined with the distance measure D , may effectively raise their awareness of their lack of time management and self-regulation skills. Another possible application may be to use this information to nudge students who do not follow the recommended pathway and provide them with regular, timely, and personalised feedback.

At course level, a dashboard providing weekly information about the mean distance between the students' pathways and the recommended pathway, could be useful to alert the instructor about a worrying students' disengagement trend. A constantly high distance value could be indicative of a flaw in the course design, for example with respect to the amount of effort required from the students. A sudden increase of the distance measure could be indicative of a particularly challenging topic which causes disengagement from the students. It could also indicate that the students have a busy schedule forcing them to make undesirable choices concerning their time and effort allocations.

Our study suffers from several limitations. First, the data was collected on a weekly basis, where daily collection would have provided more accurate time stamps. This is easy to correct, and more data will be collected for the future cohorts of students. Second, in its current formulation, the distance measure does not allow us to precisely discriminate between the different learner profiles (selective, late, irregular, etc.). One possibility, which is under investigation, would be to use different weightings for different types of activity and to vary the weightings as the semester advances. Another possibility would be to formulate different distance measures, each measure being designed to capture a particular learner profile. Better discrimination between the different types of learners will enable the formulation of more accurate and pertinent nudging.

We have a lot of data that we have not yet exploited. Future work includes calculating correlations between different types of activity (completion and scores) and course performance. We have also started to work on the design of different dashboards for students and for instructors.

5. REFERENCES

- [1] Ally, M. 2004. Foundations of educational theory for online learning. In *The theory and practice of online learning*, ed. T. Anderson, 15–44. Edmonton, CA: Athabasca University Press.
- [2] Barnard, L., Lan, W.Y., To, Y.M., Paton, V.O. and Lai, S.L. 2009. Measuring self-regulation in online and blended learning environments. *The Internet and Higher Education*, 12, 1–6.
- [3] Britton, B. K. and Tesser, A. 1991. Effects of time management and practices on college grades. *Journal of Educational Psychology*, 83, 405–410.
- [4] Broadbent, J., and Poon, W. L. 2015. Self-regulated learning strategies & academic achievement in online higher education learning environments: A systematic review. *The Internet and*

- [5] Broadbent, J. 2017. Comparing online and blended learner's self-regulated learning strategies and academic performance. *The Internet and Higher Education*, 33, 24-32.
- [6] Brockett, R. G. and Hiemstra, R. 1991. *Self-direction in Adult Learning: Perspectives on theory, Research, and Practice*, Routledge, New York, NY, USA.
- [7] Candy, P. C. 1991. *Self-direction for Lifelong Learning: A Comprehensive Guide to theory and Practice*, Jossey-Bass, San Francisco, CA, USA.
- [8] Crosslin, M. 2018. Exploring self-regulated learning choices in a customisable learning pathway MOOC. *Australasian Journal of Educational Technology*, Vol. 34 No. 1, 131-144. <https://doi.org/10.14742/ajet.375>.
- [9] Ferrari, J. R. 2001. Procrastination as self-regulation failure of performance: effects of cognitive load, self-awareness, and time limits on 'working best under pressure'. *European Journal of Personality*, 15(5), 391-406.
- [10] Garrison, D. R. 1992. Critical thinking and self-directed learning in adult education: an analysis of responsibility and control issues, *Adult Education Quarterly*, vol. 42, no. 3, pp. 136-148.
- [11] Garrison, D. R. 1997. Self-directed learning: toward a comprehensive model, *Adult Education Quarterly*, vol. 48, no. 1, pp. 18-33.
- [12] Gleadow, R., Macfarlan, B., & Honeydew, M. (2015). Design for learning - a case study of blended learning in a science unit. *F1000Research*, 4:898, 1-17.
- [13] Klassen, R. M., Krawchuk, L. L., and Rajani, S. 2008. Academic procrastination of undergraduates: Low self-efficacy to self-regulate predicts higher levels of procrastination. *Contemporary Educational Psychology*, 33(4), 915-931.
- [14] Knowles, M. S. 1975. *Self-directed Learning: A Guide for Learners and Teachers*, Association Press, New York, NY, USA.
- [15] Loeng, S. 2020. Self-Directed Learning: A Core Concept in Adult Education, *Education Research International*, vol. 2020, 1-12. <https://doi.org/10.1155/2020/3816132>
- [16] Loyens, S. M. M., Magda, J. and Rikers, S. M. M. 2008. Self-directed learning in problem-based learning and its relationships with self-regulated learning, *Educational Psychology Review*, vol. 20, no. 4, pp. 411-427.
- [17] Macan, T., Shahani, C., Dipboye, R. L., & Phillips, A. P. 1990. College students' time management: Correlations with academic performance and stress. *Journal of Educational Psychology*, 82, 760-768.
- [18] Pintrich, P. R. 1999. The role of motivation in promoting and sustaining self-regulated learning. *International Journal of Educational Research*, 31, 459-470.
- [19] Senécal, C., Koestner, R. & Vallerand, R.J. 1995. Self-Regulation and Academic Procrastination. *The Journal of Social Psychology*, 135:5, 607-619.
- [20] Tam, V., Lam, E.Y. & Fung, S.T. 2014. A new framework of concept clustering and learning path optimization to develop the next-generation e-learning systems. *J. Comput. Educ.* 1, 335-352.
- [21] Uzun, B., LeBlanc, S., & Ferrari, J. R. 2020. Relationship between academic procrastination and self-control: the mediational role of self-esteem. *College Student Journal*, 54(3), 309-316.
- [22] Wolters, C.A., Won, S. & Hussain, M. 2017. Examining the relations of time management and procrastination within a model of self-regulated learning. *Metacognition Learning* 12, 381-399.
- [23] Zacks, S. & Hen, M. 2018. Academic interventions for academic procrastination: A review of the literature, *Journal of Prevention & Intervention in the Community*, 46:2, 117-130.
- [24] Zimmerman, B. J., & Schunk, D. H. (Eds.). 2001. *Self-regulated learning and academic achievement: Theoretical perspectives*. Routledge. 336 pages. ISBN 9780805835618

Student Perception on the Effectiveness of On-Demand Assistance in Online Learning Platforms

Aaron Haim
Worcester Polytechnic Institute
100 Institute Road
Worcester, Massachusetts, USA
ahaim@wpi.edu

Neil T. Heffernan
Worcester Polytechnic Institute
100 Institute Road
Worcester, Massachusetts, USA
nth@wpi.edu

ABSTRACT

Studies have shown that **on-demand assistance**, additional instruction given on a problem per student request, improves student learning in online learning environments. Students may have opinions on whether an assistance was effective at improving student learning. As students are the driving force behind the effectiveness of assistance, there could exist a correlation between students' perceptions of effectiveness and the computed effectiveness of the assistance. This work conducts a survey asking secondary education students on whether a given assistance is effective in solving a problem in an online learning platform. It then provides a cursory glance at the data to view whether a correlation exists between student perception and the measured effectiveness of an assistance. Over a three year period, approximately twenty-two thousand responses were collected across nearly four thousand, four hundred students. Initial analyses of the survey suggest no significance in the relationship between student perception and computed effectiveness of an assistance, regardless of if the student participated in the survey. All data and analysis conducted can be found on the Open Science Foundation website¹.

Keywords

Online Education, On-Demand Assistance, Student Perception

1. INTRODUCTION

Collecting student perceptions on educators' performances, practices, and generated content has been well established in higher education, along with its ability to potentially improve student instruction [3, 10]. However, very few have been conducted on student perception in secondary education settings [11]. Additionally, surveys conducted on student perception rarely consider individual pieces of content and their effectiveness in context.

¹<https://osf.io/f8w9p/>

A. Haim and N. Heffernan. Student perception on the effectiveness of on-demand assistance in online learning platforms. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 734–737, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853053>

On-demand assistance generally improves student leaning [8, 12, 17, 20] in online learning platforms. Educators and their assistance are evaluated through computations to maintain or improve the level of quality and effectiveness [13, 16]. As such, student perceptions on assistance effectiveness are rarely polled, leaving a gap in their evaluation.

In 2017, ASSISTments, an online learning platform [9], deployed the Special Content System, formerly known as TeacherASSIST. The Special Content System allows educators to create and provide on-demand assistance for problems assigned to their students. On-demand assistance was known as *student-supports* within the application, with most as either hints or explanations. Additionally, educators marked as *star-educators* had their *student-supports* provided to any student, regardless of their class, for any problem a class's educator did not generate a *student-support* for. The effectiveness of a *student-support* was evaluated using whether the student answered the next problem in the problem set correctly on their first try [13, 16].

In the Fall semester of 2018, the Special Content System, on providing *student-supports* to the students, would prompt the student to provide feedback on the helpfulness of the *student-supports* as shown in Figure 1. The student may have chosen to evaluate the *student-support* by clicking the "Yes" or "No" button or not responded by completing and moving to the next problem. The feature was disabled in 2020 within the new, redesigned ASSISTments platform; however, educators who have not migrated from the original platform will still be provided this feature: prompting students on the helpfulness of a given *student-support*.

The first part of this work will analyze the students' perceptions of helpfulness and compare them to the overall effectiveness of the *student-support*. If correlated, students could be effective in filtering *student-supports*. In online learning environments which crowdsource their on-demand assistance [13, 12, 17], it may improve automation and independence from moderating bodies. The second part of this work will analyze the students' perceptions of helpfulness and compare them to the responding students' effectiveness of the *student-support*. This may give stronger claims towards a student's ability to predict the effectiveness of assistance provided to themselves.

In summary, this work aims to answer the following research

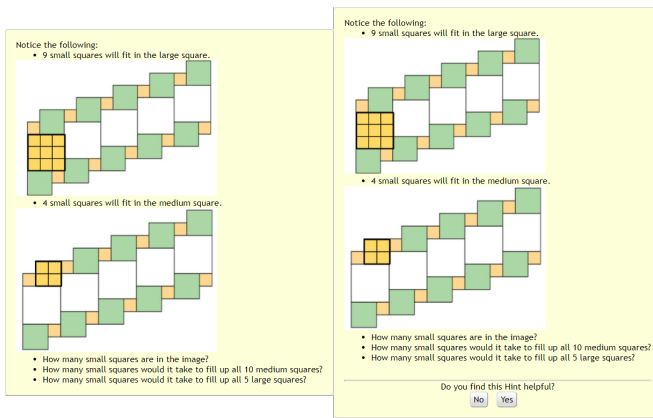


Figure 1: A *student-support* provided to the student (left) prompting the student for their perception on its helpfulness (right).

questions:

1. Can student perceptions accurately predict the overall effectiveness of *student-supports*?
2. Can student perceptions accurately predict the effectiveness of *student-supports* for students who participated in the survey?

2. BACKGROUND

In this work, ASSISTments will be used to conduct the studies. ASSISTments² is a free, online learning platform providing feedback to inform educators on classroom instruction [9]. ASSISTments provides open source curricula, the majority of which is K-12 mathematics, containing problems and assignments that teachers can assign to their students. Students complete assigned problems in the ASSISTments Tutor. For nearly all problem types, students receive immediate feedback on response submission, which tells the student the correctness of the answer [6]. When a *student-support* has been written for a problem, a student can request to receive a *student-support* during the span of problem completion. *Student-supports* may come in the form of hints which explain how to solve parts of the problems [8, 17], similar problem examples [12], erroneous examples [17, 1], and full solutions to the problems [18, 19].

ASSISTments runs an application known as the Special Content System in which *student-supports*, usually as hints or explanations, are crowdsourced from educators. *Student-supports* are created by educators and then provided to students when solving. In this work, data from the ASSISTments Dataset generated by the Special Content System [14, 15] alongside the student perception on *student-support* effectiveness is used to analyze whether student perceptions accurately predict the effectiveness of *student-supports*.

3. METHODOLOGY

Data was collected within the original ASSISTments platform over the course of three years. During this time period,

²<https://assistments.org/>

when a *student-support* was provided to a student, a prompt may appear asking about the effectiveness of the *student-support*, as shown in Figure 1. The student additionally had an equally likely chance that a prompt may not be shown. 46,620 responses across 13,509 students were recorded on the effectiveness of a *student-support*. Preprocessing on the responses removed any where the effectiveness of a *student-support* could not be determined. Additionally, any students who only answered one of the available choices across many problems were discarded as the provided answers may not be in full consideration of the question or without major bias. After processing, 21,736 responses across 4,374 students remained. The responses consisted of 5,861 *student-supports* across 4,120 problems.

After the response were processed, they were grouped by problem and sub-grouped further by *student-support*. Each sub-group of *student-supports* contained the data for the student perception of effectiveness and the effectiveness data as specified by the research questions: overall effectiveness and the effectiveness across the students who responded respectively. For every pair of *student-supports* with a problem group, a Sign Test [5] was performed between the student perception of effectiveness and measured effectiveness of the *student-supports*. As A Sign Tests measures whether the differences between pair of data is consistent, the results are expected to follow a binomial distribution. As such, a two-sided Binomial Test [4] was performed to measure whether the student perception of effectiveness and the measured effectiveness is consistent. If there was less than two *student-supports* available for a problem or the difference between the perception or measured effectiveness was zero, then no useful comparison could be made and was thus discarded [5]. The results of the pairwise comparisons between all *student-supports* are reported using the *Pair* prefix within Table 1. To make claims on the effectiveness of *student-supports* on a given problem, another two-sided Binomial Test was performed only on the two supports with the largest combined sample size. The results of the pairwise comparison between the two *student-supports* with the largest sample size per problem are reported using the *Problem* prefix within Table 1. As multiple analyses were being performed on the dataset, the Benjamini-Hochberg Procedure [2] was applied to the corresponding p-values to reduce the false discovery rate.

To measure the effectiveness across only those who participated in the survey, a Chi-squared Test [7] was performed comparing the relationship between the student perception of effectiveness and the measured effectiveness across students who responded for a *student-support* in addition to the Sign Test. In comparison to the Sign Test which measures consistent differences, a Chi-Squared Test measures the difference in frequencies between two relationships. A Chi-squared Test was not conducted on the overall effectiveness as the sample size considers all students who were provided the associated *student-support* which violates the equality of sample sizes across observation. As the results of all the Chi-Squared Tests performed is infeasible to report, only those which found significant results are shown in Table 2.

4. RESULTS

Table 1: Binomial Test of the Sign Tests on student perception of effectiveness and the measured effectiveness of *student-supports*.

Test Performed	Sample Size	Number of Successful Sign Tests	P-Value	Corrected P-Value
Pair: Overall	1,699	876	0.2071	0.379
Problem: Overall	1,182	607	0.3672	0.379
Pair: Responding Students	1,320	695	0.0575	0.23
Problem: Responding Students	942	485	0.379	0.379

4.1 Overall Effectiveness

As shown in Table 1, across the 5,861 available *student-supports* with responses, 1,699 pairs of *student-supports* were created with 876 pairs (proportion estimate of 0.5156) returning a correlated difference on the Sign Test for overall effectiveness. Across the 1,699 pairs of *student-supports*, there were 1,182 problems, of which 607 (proportion estimate of 0.5135) had a correlated difference on the Sign Test. Both results were not significant ($\alpha < 0.05$) before Benjamini-Hochberg correction.

4.2 Effectiveness Across Responding Students

Also in Table 1, 1,320 pairs of *student-supports* were created with 695 pairs (proportion estimate of 0.5265) returning a correlated difference on the Sign Test for effectiveness across responding students. Across the 1,320 pairs of *student-supports* there were 942 problems, of which 485 (proportion estimate of 0.5149) had a correlated difference on the Sign Test. Both results were not significant ($\alpha < 0.05$) before Benjamini-Hochberg correction.

The Chi-squared Test ($z_{0.05} \geq 3.8415$) revealed only six out of the 5,861 responded *student-supports* with a relationship between the student perception of effectiveness and the measured effectiveness across the responding students. The results of the significant Chi-squared Tests are shown in Table 2. The significance of the results can be attributed to randomness due to sample size ($6/5861 = 0.001 < 0.05$). The corrected p-values using the Benjamini-Hochberg Procedure[2] also conveys the non-significance of the results at $p = 1$.

5. CONCLUSION

In this work, initial findings show no significant relationship between student perception of effectiveness and the measured effectiveness of a *student-support*. Only 5% of students who used the original system were considered in the performed analysis. Additionally, less than 1% of the responses collected showed any significance in the relationship. As such, the significance of any current results can be attributed to randomness. The response set is unlikely to be generalizable to other online learning platforms or across students and problems.

Potential improvements to further analysis could view the features of the participating students or problems, such as prior knowledge or problem accuracy. The prompts could also be subdivided further to determine if phrasing has a desired effect on student perception. Accurate markers of student response times could also be recorded to differentiate responses before and after completing the given problem. Additionally, the location of when the prompt was delivered

could be moved to better reflect the measure of effectiveness used for an on-demand assistance.

Although no evidence of the relationship between student perception of effectiveness and the measured effectiveness of a *student-support* was found in this work, other qualities could attempt to better provide understanding of student perceptions on effectiveness in more granular surveys. Future work can explore better opportunities in student perceptions on effectiveness of *student-supports* for themselves. Afterwards, further steps can be taken to gradually generalize effectiveness overall and eventually to other online learning platforms.

6. ACKNOWLEDGMENTS

We would like to thank the NSF (e.g., 2118725, 2118904, 1950683, 1917808, 1931523, 1940236, 1917713, 1903304, 1822830, 1759229, 1724889, 1636782, & 1535428), IES (e.g., R305N210049, R305D210031, R305A170137, R305A170243, R305A180401, & R305A120125), GAANN (e.g., P200A180088 & P200A150306), EIR (U411B190024 & S411B210024), ONR (N00014-18-1-2768), and Schmidt Futures. None of the opinions expressed here are that of the funders. We are funded under an NHI grant (R44GM146483) with Teachly as a SBIR.

7. REFERENCES

- [1] D. M. Adams, B. M. McLaren, K. Durkin, R. E. Mayer, B. Rittle-Johnson, S. Isotani, and M. Van Velsen. Using erroneous examples to improve mathematics learning with a web-based tutoring system. *Computers in Human Behavior*, 36:401–411, 2014.
- [2] Y. Benjamini and Y. Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal statistical society: series B (Methodological)*, 57(1):289–300, 1995.
- [3] J. A. Centra. Student ratings of instruction and their relationship to student learning. *American Educational Research Journal*, 14(1):17–24, 1977.
- [4] W. G. Cochran. The efficiencies of the binomial series tests of significance of a mean and of a correlation coefficient. *Journal of the Royal Statistical Society*, 100(1):69–73, 1937.
- [5] W. J. Dixon and A. M. Mood. The statistical sign test. *Journal of the American Statistical Association*, 41(236):557–566, 1946.
- [6] M. Feng and N. T. Heffernan. Informing teachers live about student learning: Reporting in the assistent system. *Technology Instruction Cognition and Learning*, 3(1/2):63, 2006.

Table 2: Chi-Squared Tests with a relationship between the student perception of effectiveness and the measured effectiveness across the responding students for a *student-support*. The tables are in the following order: both student perception and measured effectiveness are true, only student perception is true, only measured effectiveness is true, neither student perception and measured effectiveness are true.

Contingency Table		Expected Table		Critical Statistic	P-Value
9	2	6.05	4.95	4.8995	0.0269
2	7	4.95	4.05		
3	14	6.12	10.88	5.4767	0.0193
6	2	2.88	5.12		
2	10	4.67	7.33	4.9383	0.0263
5	1	2.33	3.67		
11	5	7.68	8.32	5.5314	0.0187
1	8	4.32	4.68		
7	2	4.05	4.95	4.8995	0.0269
2	9	4.95	6.05		
14	3	9.71	7.29	8.7625	0.0031
2	9	6.29	4.71		

- [7] T. M. Franke, T. Ho, and C. A. Christie. The chi-square test: Often used and more often misinterpreted. *American Journal of Evaluation*, 33(3):448–458, 2012.
- [8] N. T. Heffernan and C. L. Heffernan. The assistments ecosystem: Building a platform that brings scientists and teachers together for minimally invasive research on human learning and teaching. *International Journal of Artificial Intelligence in Education*, 24(4):470–497, 2014.
- [9] N. T. Heffernan and C. L. Heffernan. The assistments ecosystem: Building a platform that brings scientists and teachers together for minimally invasive research on human learning and teaching. *International Journal of Artificial Intelligence in Education*, 24(4):470–497, 2014.
- [10] J. W. B. Lang and M. Kersting. Regular feedback from student ratings of instruction: Do college teachers improve their ratings in the long run? *Instructional Science*, 35(3):187–205, May 2007.
- [11] L. Mandouit. Using student feedback to improve teaching. *Educational Action Research*, 26(5):755–769, 2018.
- [12] B. M. McLaren, T. van Gog, C. Ganoë, M. Karabinos, and D. Yaron. The efficiency of worked examples compared to erroneous examples, tutored problem solving, and problem solving in computer-based learning environments. *Computers in Human Behavior*, 55:87–99, 2016.
- [13] T. Patikorn and N. T. Heffernan. Effectiveness of crowd-sourcing on-demand assistance from teachers in online learning platforms. In *Proceedings of the Seventh ACM Conference on Learning @ Scale, L@S '20*, page 115–124, New York, NY, USA, 2020. Association for Computing Machinery.
- [14] E. Prihar, A. F. Botelho, R. Jakhmola, and I. Heffernan, Neil T. Assistments 2019-2020 school year dataset, Dec 2021.
- [15] E. Prihar and M. Gonsalves. Assistments 2020-2021 school year dataset, Nov 2021.
- [16] E. Prihar, T. Patikorn, A. Botelho, A. Sales, and N. Heffernan. Toward personalizing students’ education with crowdsourced tutoring. In *Proceedings of the Eighth ACM Conference on Learning @ Scale, L@S '21*, page 37–45, New York, NY, USA, 2021. Association for Computing Machinery.
- [17] L. M. Razzaq and N. T. Heffernan. To tutor or not to tutor: That is the question. In V. Dimitrova, editor, *AIED*, pages 457–464. IOS Press, 2009.
- [18] J. Whitehill and M. Seltzer. A crowdsourcing approach to collecting tutorial videos—toward personalized learning-at-scale. In *Proceedings of the Fourth (2017) ACM Conference on Learning@ Scale*, pages 157–160, 2017.
- [19] J. J. Williams, J. Kim, A. Rafferty, S. Maldonado, K. Z. Gajos, W. S. Lasecki, and N. Heffernan. Axis: Generating explanations at scale with learnersourcing and machine learning. In *Proceedings of the Third (2016) ACM Conference on Learning@ Scale*, pages 379–388, 2016.
- [20] D. Wood, J. S. Bruner, and G. Ross. The role of tutoring in problem solving. *Child Psychology & Psychiatry & Allied Disciplines*, 17(2):89–100, 1976.

Theory-Informed Problem-Solving Sequential Pattern Visualization

Zilong Pan

The University of Texas at Austin
panzi89@utexas.edu

Min Liu

The University of Texas at Austin
mliu@austin.utexas.edu

ABSTRACT

This study applied a problem-based learning framework to visualize students' problem-solving patterns using sequential log data. Comparing the theory-informed visualization with a graph without theoretical guidance proved that problem-solving visualizations with a theoretical foundation were more interpretable for researchers and educators. The presentation of the graph was more explicit, and the outcomes can be generalizable to other related PBL studies. Besides, the theory-informed visualization can also be used by teachers to provide differentiated scaffoldings to support different groups of students when facilitating problem-based learning activities.

Keywords

Learning analytics, problem-based learning, theory-informed, data visualization.

1. BACKGROUND & INTRODUCTION

Problem-based learning (PBL) is an instructional method in which students learn through facilitated problem solving that centers on an ill-structured problem [8]. It is a constructivist educational strategy and should be performed via a student-centered pedagogical approach [6]. In other words, when engaging in a PBL activity, learners should take a more active role in obtaining knowledge and skills [9]. Instructors need to serve as facilitators who provide individualized instructions to accommodate students' diverse progress [17].

In a constructivist activity like PBL, students would be at different paces and engage in various tasks. Thus, to better facilitate students' problem-solving, it is vital to monitor their progress [6]. Only after educators or researchers tracked students' different progress successfully and precisely, then they could provide individualized instructions to facilitate students [13].

The recent advancement of virtual PBL environments provided researchers opportunities to track students' problem-solving progress [3]. When students were engaging with virtual PBL environments, a large number of usage clickstream log data would be generated. The clickstream log data is time-stamped and captures students' fine-grained behaviors [18]. Researchers and educators can employ these data to monitor students' real-time progress or to provide just-in-time interventions.

Z. Pan and M. Liu. Theory-informed problem-solving sequential pattern visualization. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 738–742, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853059>

The collected log data is often large amounts and consists of multiple features including student ID, timestamp including start time, end time, the name of accessed tool, tool use actions such as tool open or close, and so on. Collecting the clickstream log is just the first step, a more crucial step is how researchers and educators process the log data to make it interpretable for further educational practices. In this case, educational theories are needed in guiding researchers to process the collected log data and make it interpretable and relevant to educators.

2. PURPOSE OF STUDY

The research involving collecting, analyzing, and reporting learners' usage log data is considered learning analytics studies (LA). Researchers have been exploring and refining the approaches of incorporating educational theories when conducting LA studies [2]. For instance, a study [7] incorporated the framework of engagement in processing student-generated log data on MOOCs. Researchers applied exploratory and confirmative factor analysis upon collected log data, and related the outcome factors to three engagement constructs: affective, cognitive, and behavioral engagement. This theory-informed data processing allowed researchers better to interpret the association between usage logs and learner performance. For example, learners' video watching duration is associated with final scores because they were positively correlated with the behavioral and cognitive engagements. In another study [15], researchers mapped usage logs on learning management systems to self-regulated learning (SRL) phases. For instance, students' actions of accessing objectives or lesson overviews were associated with the Forethought phase of SRL. These actions reveal the traces of learners' goal-setting behaviors, an important component of the forethought phase. Following the SRL framework, researchers could draw a clear path between nuanced learning analytics features and the nuances of learning theories. In return, the framework provided a solid theoretical foundation for further instructional practices and the future designs of the learning platforms. Although researchers have taken initiatives of involving theories to process log data generated from various educational platforms. However, seldom studies have focused on incorporating theories to process problem-solving behavioral logs. Although previous studies applied LA techniques to analyze problem-solving logs [3, 12], however, to produce inferable and generalizable research findings for future PBL research, more theory-based log processing approaches are needed.

In addition, data visualization was widely applied in representing log data outcomes, especially when the amount of data is large [11]. LA studies can be benefited from data visualization techniques is because the visualizations can represent the large amount of data in a compact format without losing essential information [1]. For example, a study [4] extracted student log data from a virtual geometry game to visualize their problem-solving patterns. These graphs helped researchers to examine students' problem-solving patterns explicitly, they were able to see whether students were on

the expected path during the activity. In another study [14], researchers used area graphs to represent students' different tool usage frequencies in a virtual PBL environment. These graphs allowed researchers to efficiently identify the different usage patterns between different groups of students. Using log data to create data visualizations brings researchers larger flexibility to examine user behavior patterns from different aspects. The outcomes of the visualizations could also be used in the classroom to inform teachers about providing just-in-time supports [16].

Considering the advantage of data visualization and the need to incorporate educational theories in processing log data, especially in the PBL context. The purpose of this study is to merge the gap by providing a theory-informed LA method to process sequential behaviors generated in a PBL activity. Then, by visualizing the sequential outcomes, the ultimate goal of proposing this approach is to (1) generate theory-based interpretable sequential patterns for researchers and (2) assist teachers in being better informed about students' progress to support their PBL facilitation.

3. METHOD

3.1 Research Context

The PBL environment applied in this study is called Alien Rescue (AR). In this environment, students play the role of scientists to help six alien species to relocate to our solar system. By solving the problem of which species can survive on which planet or moons in our solar system, students would gain related scientific knowledge aligned with 6th-grade science standards.

Figure 1 presents the screenshots of the AR environment. It is a 3D immersive problem-solving platform, and there are ten different cognitive tools embedded for students to use. These tools, like the Solar system database or Notebook, can provide students with needed information and scaffoldings to find the solutions.



Figure 1. Alien Rescue PBL environment

3.2 Participants and Data Source

A total of 114 six grade students engaged with the AR PBL environment through 15 class sessions. More specifically, 61 students conducted this activity virtually at home, and 53 students conducted this activity in-person at school.

In all, 45554 lines of behavioral log data generated by participants were collected for data processing and analysis. The features of the collected log data include:

- Student ID.
- Activity start time (by second).
- Activity end time (by second).
- The name of the accessed tool.
- Tool usage actions such as tool open or close.

3.3 Theory-Informed Data Processing

To track and analyze students' problem-solving behaviors and present generalizable outcomes, the framework of PBL process was applied [8]. Based on the framework, problem-solving usually consists of multiple phases, including identifying facts (IF) and Knowledge deficiency (IKD), generating hypotheses (GH), and

Solutions (GS). Learners perform the phases iteratively or in different order until solutions are produced. To better interpret students' problem-solving patterns using log data and related to the framework, 28 different kinds of behavioral log actions in AR were connected with several problem-solving phases based on the concept of each phase [8]. Table 1 presents an example of the alignment. The first phase in the framework is the activities for students to understand the problem scenario. AR environment provides several tools such as Alien or Concept Databases for students to collect the information they need for solving the problem. Thus, the access to these tools aligns with the idea of Identify Facts (IF) phase. Plus, students needed to generate possible hypotheses as they gathered more information. The probe sending feature in AR is a critical component that students can actively obtain information to generate hypotheses. Thus, the actions related to sending probes were aligned with Generate Hypothesis (GH) phase. Moreover, AR provided students with a Notebook tool to organize collected information and identify missing pieces, which resonates with the phase Identify Knowledge Deficiency (IKD). Plus, Notebook also allows students to generate possible solutions via a Comparing Notes feature. This feature lets students conveniently compare the information they obtained in the environment to decide which place is suitable for which alien. In this case, actions relevant to comparing notes were aligned with Generate Solution (GS) phase.

Table 1. Examples for alignment process

Log Activities in AR	Phases in PBL Cycle	Definition
Alien Database: Click Concept Database: Click	Identify Facts (IF, index = 1)	Students identify the relevant facts from the environment, which helps them represent the problem.
Probe Design: Change Probe Name; Click Back Button	Generate Hypothesis (GH, index = 2)	Students take the initiative to generate possible hypotheses as they understand the problem better.
Notebook: Click; Create; Delete; Edit	Identify Knowledge Deficiency (IKD, index = 3)	Students identify the knowledge that is gathered against knowledge that is still needed.
Notebook: Compare; Edit	Generate Solutions (GS, index = 4)	Students use the obtained knowledge to produce solutions.

Ultimately, all log data were transformed into 1789 sequences that reflect students' problem-solving sequential patterns. Table 2 presents an example of processed outcomes. For instance, the first row indicates the sequence performed by student 1 in May. 25th is IF, IKD, GH, IF, IKD, and GS, which is different from the sequence the student conducted in May. 26th. These sequences were then used for sequential analysis and creating visualizations to examine student problem-solving patterns.

Table 2. Example of processed sequences

Student	Date	Activity Sequence
Student 1	May.25th	[1, 3, 2, 1, 3, 4]
Student 1	May.26th	[1, 3, 1, 3, 1, 4, 1, 4, 1]
Student 2	May.25th	[1, 2, 1, 3, 2, 1, 3, 1, 2, 3]

Note. 1 = Identify Facts (IF), 2 = Generate Hypothesis (GH), 3 = Identify Knowledge Deficiency (IKD), 4 = Generate Solutions (GS)

3.4 Data Visualization Techniques

Problem-solving is a sequential activity that consists of multiple phases such as Identify Facts or Generate Hypothesis. The paths students go through during these phases are not always linear. In fact, during a PBL activity, students usually go iteratively or circle among different problem-solving phases [8]. The Sankey graph is a type of graph that symbolizes sequential activities. It can mimic users' iterative sequential behaviors. More importantly, the graph can also present the strength of each sequential path: the wider the path between two phases, the larger proportion of the sequential activity between two phases among all sequences in the graph. Thus, Sankey graphs were used to visualize the iterative manner of students' problem-solving efforts.

4. RESULTS & DISCUSSION

4.1 Theory-informed visualization

Figure 2 was created to represent students' problem-solving behavioral patterns. For instance, the label 20% on the top of the figure means, among all the paths performed by students, 20% of the sequential actions were from Identify Knowledge Deficiency (IKD) to Identify Facts (IF) phase.

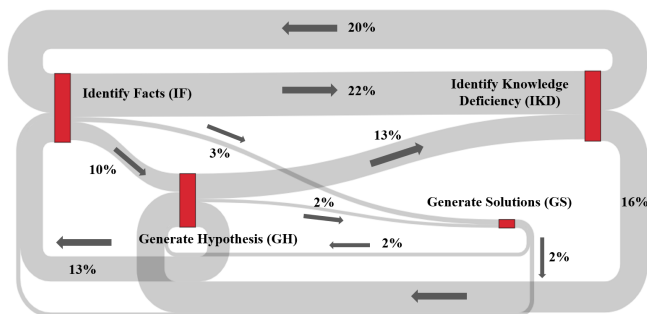


Figure 2. Theory-informed problem-solving patterns

Based on Figure 2, following highlights can be extracted:

- Relatively large proportions of the sequential behaviors were conducted between IKD and IF phases (20%, 22%). It means students were inclined to keep seeking information until they figured out what information was obtained and what was missing, and they would find out the ones they needed.
- The paths involving Generate Solutions (GS) were relatively smaller among all the paths (e.g., IF to GS is 3%; GH to GS is 2%). These outcomes indicate GS is more of an end of problem-solving activity instead of a mean. It is aligned with the PBL environment setting. In the GS phase, students just needed to submit their solutions to a console in the environment, the action itself does not enhance or intervene in their problem-solving progress. In total, students are required to compose six solutions; that's why the GS phase paths are smaller. To be noted, there were some paths coming out from GS phase; it means students were reconfirming their solutions either by going back to IF (2%) to examine some information, or to GH phase (2%) to refine their hypothesis.
- The paths among IF, GH, and IKD occupy medium proportions compared to the above two findings. For instance, 16% of the actions were from IKD to GH. GH phases involve the actions that students were sending probes to narrow down their hypothesized solutions. Students may or may not have a

solution ready after the GH phase, which explains why the proportion of paths from GH to GS is relatively small (2%).

In all, figure 2 presents the problem-solving sequential patterns performed by students during a PBL activity. This graph provided an overview of how students conducted problem-solving in an iterative manner across multiple PBL phases. More importantly, this graph provided a theory-informed representation of the sequential problem-solving patterns generated by a group of middle school 6th-graders in a science PBL activity. Because the paths presented the actions transiting across different PBL phases, this outcome would be informative for other PBL research that is either in a similar or a different educational context.

In fact, to examine the benefits of theory-informed visualizations, Figure 3 was created by the authors for providing a contrast to figure 2. Figure 3 is also a Sankey graph but without aligning the problem-solving actions and the PBL phases. All nodes presented were different tools provided in the PBL environment, and all paths were students' sequential paths across each tool. For instance, from this figure, we can find out relatively large proportions of sequential actions were performed among Notebook, Solar System Database, and Alien Database. These three tools are all essential features provided in the environment for students to solve the problem. Indeed, it provided a fine-grained visualization of how students transited across different tools in the environment. However, figure 3 could be less interpretable than figure 2 in the following two aspects.

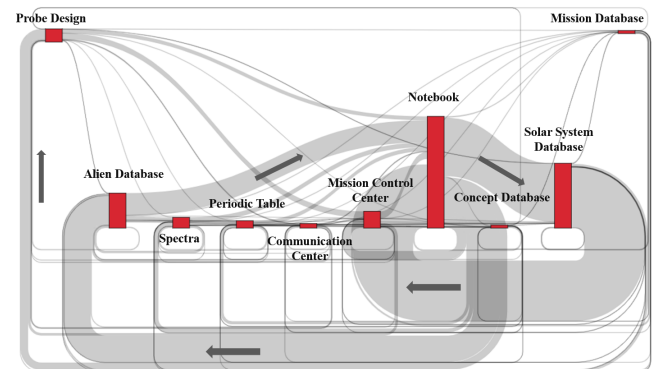


Figure 3. Patterns without PBL framework alignment

The first aspect is information presentation. In the AR PBL environment, there are a total of 10 different tools. A graph that presents students' problem-solving patterns in such an environment should incorporate all the actions among these ten tools. Compared to figure 2, which only contains four nodes, figure 3 brings a larger amount of information or cognitive load for readers to process [1]. Plus, the presentation of this graph is already congested even without the authors inputting direction and percentage for each path. In fact, many current PBL environments, such as River City [10] or Crystal Island [5], involve more than ten tools or features that students need to access. Even though a fine-grained graph that includes all tools can reveal detailed sequential behaviors and is informative to readers familiar with the environment, it might be challenging for both researchers and readers to interpret those less acquainted.

The second aspect is generalizability. A fine-grained graph like figure 3 can indeed reveal exact sequential action performance by students. For example, after accessing Probe Design, the figure shows students would engage with a series of tools such as Alien Database or Spectra. However, what do these actions mean in terms

of problem-solving patterns in general? It may be more informative for developers of this environment than researchers conducting PBL studies in different contexts. Whereas for figure 2, since the Probe Design is aligned with the GH phase (see Table 1), the figure shows 13% of actions afterward were to IDF, and 13% were to IF. These outcomes provide a more explicit picture of how students performed their PBL paths and strategies, since it is aligned with the PBL framework [8]. Other studies that also applied this framework could compare and contrast the outcomes with this study and draw inferences. In addition, researchers or educators who are contemplating conducting PBL studies or activities could pick up the information that depicts students' problem-solving patterns easily from the theory-informed graphs to apply to their future PBL design or activities.

4.2 Uses of the theory-informed visualization

Expect for interpretation, the theory-informed visualizations also present in-depth outcomes for researchers and educators about students' problem-solving patterns based on conditions. Since participants engaged in the AR PBL environment under virtual or in-person modes, the theory-informed visualizations can also provide a comparison between these two groups.

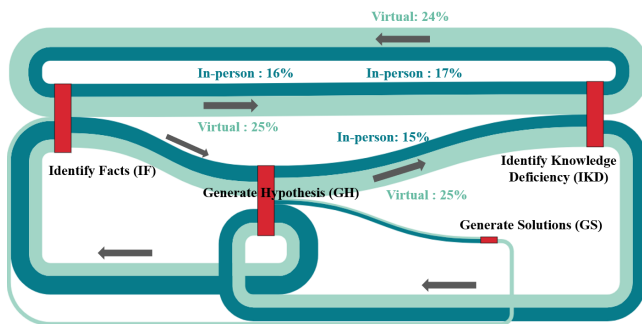


Figure 4. Problem-solving patterns grouped by learning modes

A Sankey graph was made with light blue representing the paths conducted by students in the virtual mode and dark blue representing the paths by students in the in-person mode (see Figure 4). The statistical results showed that students in the virtual mode ($M = 106$, $M = 97$) performed significantly more paths from Identify Facts (IF) to Identify Knowledge Deficiency (IKD) phase ($U = 901.00$, $z = 3.86$, $p < .01$), and from Identify Knowledge Deficiency (IKD) back to Identify Facts (IF) phase ($U = 829.00$, $z = 4.28$, $p < .01$) than their in-person mode peers ($M = 82$, $M = 72$), these statistical outcomes are resonated with the Sankey graph that light blue path is wider than dark blue paths. Moreover, as indicated in Figure 4, students in the virtual mode ($M = 8$) were more inclined to use the generated hypothesis to assist their knowledge deficiency identification process ($U = 879.50$, $z = 2.16$, $p < .05$) than students in the in-person mode ($M = 12$). Based on these results, the students in the virtual mode appeared to be more careful when identifying knowledge deficiency by organizing their collected information on Notebook. They were more likely to collect information from different databases (ID) or draw results from the probes they sent (GH) first, then input the information on Notebook for further uses. In fact, carefully identifying sufficient information from different databases and collecting returned data from probes are expected problem-solving sequential paths that would provide students with better solution outcomes [8].

In addition, the graph also showed that students in the in-person mode did not perform the paths from Generate Solutions (GS) to Identify Facts (IF), the dark blue path is missed between these two

phases. To conceptualize and finalize robust solutions, it would be helpful for students to check with different databases to ensure the correct information supported the solution. Therefore, the above outcomes indicate that students in the virtual mode would perform the problem-solving actions close to the expected path, leading to better problem-solving outcomes. These results presented that students under different learning modes performed distinctive problem-solving strategies. It indicates that when teachers facilitate PBL activities, different scaffoldings instructions should be provided based on students' learning mode. For example, teachers can encourage in-person students to identify what information they still need before generating hypothesis, or remind them to examine the facts before submitting their final solutions.

In addition, the gender variable can also be incorporated by the theory-informed visualization. Figure 5 was made to visualize the different problem-solving sequential patterns between genders. The light blue color represents the paths conducted by male students, and the dark blue color represents female students. It is noticeable that from Identify Facts (IF) to Generate Hypothesis (GH) phase, the light blue path is wider than the dark blue path. A similar pattern could be observed from Generate Hypothesis (GH) back to Identify Facts (IF) phase as well. Statistical results showed that under condition A, male students ($M = 13$, $M = 17$) performed significantly more paths from Identify Facts to Generate Hypothesis (GH) phase ($U = 1027.00$, $z = 3.14$, $p < .01$), and from Generate Hypothesis back to Identify Facts (IF) phase ($U = 1137.00$, $z = 2.50$, $p < .05$) than their female peers ($M = 8$, $M = 12$). These results indicate that male students dedicated more efforts to generating hypotheses directly after collecting information from different databases (see Table 1), and these actions were presented in a repeated manner. On the other hand, female students might be more cautious when generating hypotheses, as indicated on the graph, female students (dark blue) were more likely to generate hypotheses after identifying knowledge deficiency than male students (light blue). Although this path is not significantly different by gender, it still reveals nuanced sequential differences between genders. Therefore, teachers can provide scaffoldings to different students based on gender accordingly. For instance, teachers can remind male students not to make prompt hypotheses right after identifying facts from databases. But instead, they can check what information is missing and what they already have to draw better hypotheses.

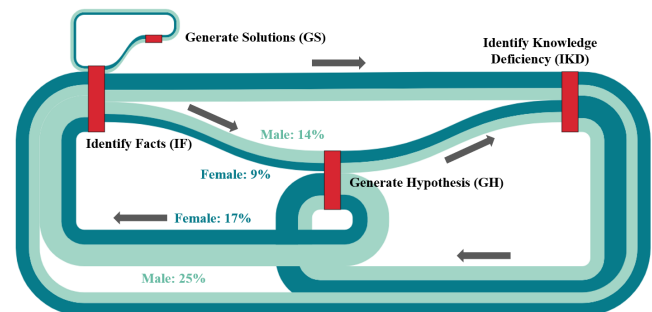


Figure 5. Problem-solving patterns grouped by genders

5. CONCLUSION

In conclusion, this study's findings revealed that theory-informed visualizations could provide more interpretable outcomes when examining students' sequential activities during problem-solving activities. The overall presentation of the visualization would be more explicit and generalizable to other relevant studies.

Moreover, the theory-informed visualizations can also demonstrate students' patterns based on different conditions such as learning modes or genders. These nuanced outcomes can inform teachers' differentiated scaffoldings when facilitating PBL activities.

6. REFERENCES

- [1] Alhadad, S.S.J. 2018. Visualizing Data to Support Judgment, Inference, and Decision Making in Learning Analytics: Insights from Cognitive Psychology and Visualization Science. *Journal of Learning Analytics*. 5, 2 (Aug. 2018), 60-85. DOI:<https://doi.org/10.18608/jla.2018.52.5>.
- [2] de Barba, P.G., Kennedy, G.E. and Ainley, M.D. 2016. The role of students' motivation and participation in predicting performance in a MOOC: Motivation and participation in MOOCs. *Journal of Computer Assisted Learning*. 32, 3 (Jun. 2016), 218-231. DOI:<https://doi.org/10.1111/jcal.12130>.
- [3] Belland, B.R. 2017. *Instructional Scaffolding in STEM Education*. Springer International Publishing.
- [4] Cecon Ribeiro, P., Biles, M.L., Lang, C., Silva, C. and Plass, J.L. 2018. Visualizing log-file data from a game using timed word trees. *Information Visualization*. 17, 3 (Jul. 2018), 183-195. DOI:<https://doi.org/10.1177/1473871617720810>.
- [5] Cloude, E.B., Dever, D.A., Wiedbusch, M.D. and Azevedo, R. 2020. Quantifying Scientific Thinking Using Multichannel Data With Crystal Island: Implications for Individualized Game-Learning Analytics. *Frontiers in Education*. 5, (Nov. 2020), 572546. DOI:<https://doi.org/10.3389/educ.2020.572546>.
- [6] English, M.C. and Kitsantas, A. 2013. Supporting Student Self-Regulated Learning in Problem- and Project-Based Learning. *Interdisciplinary Journal of Problem-Based Learning*. 7, 2 (Sep. 2013), 128-150. DOI:<https://doi.org/10.7771/1541-5015.1339>.
- [7] Fincham, E., Whitelock-Wainwright, A., Kovanović, V., Joksimović, S., van Staalduinen, J.-P. and Gašević, D. 2019. Counting Clicks is Not Enough: Validating a Theorized Model of Engagement in Learning Analytics. *Proceedings of the 9th International Conference on Learning Analytics & Knowledge* (Tempe AZ USA, Mar. 2019), 501-510.
- [8] Hmelo-Silver, C.E. 2004. Problem-Based Learning: What and How Do Students Learn? *Educational Psychology Review*. 16, 3 (Sep. 2004), 235-266. DOI:<https://doi.org/10.1023/B:EDPR.0000034022.16470.f3>.
- [9] Hung, W. 2011. Theory to reality: a few issues in implementing problem-based learning. *Educational Technology Research and Development*. 59, 4 (Aug. 2011), 529-552. DOI:<https://doi.org/10.1007/s11423-011-9198-1>.
- [10] Ketelhut, D.J., Clarke, J. and Nelson, B.C. 2010. The Development of River City, a Multi-User Virtual Environment-Based Scientific Inquiry Curriculum: Historical and Design Evolutions. *Designs for Learning Environments of the Future*. M.J. Jacobson and P. Reimann, eds. Springer US. 89-110.
- [11] Liu, M., Kang, J., Zilong, P., Zou, W. and Lee, H. 2017. Exploring Data Visualization as an Emerging Analytic Technique. *Proceedings of E-Learn: World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education 2017* (Vancouver, British Columbia, Canada, Oct. 2017), 1681-1690.
- [12] Liu, M., Li, C., Pan, Z. and Pan, X. 2019. Mining big data to help make informed decisions for designing effective digital educational games. *Interactive Learning Environments*. 27 (Jul. 2019), 1-21. DOI:<https://doi.org/10.1080/10494820.2019.1639061>.
- [13] Liu, M., Shi, Y., Pan, Z., Li, C., Pan, X. and Lopez, F. 2021. Examining middle school teachers' implementation of a technology-enriched problem-based learning program: Motivational factors, challenges, and strategies. *Journal of Research on Technology in Education*. 53, 3 (Jul. 2021), 279-295. DOI:<https://doi.org/10.1080/15391523.2020.1768183>.
- [14] Liu, S. and Liu, M. 2020. The impact of learner metacognition and goal orientation on problem-solving in a serious game environment. *Computers in Human Behavior*. 102, (Jan. 2020), 151-165. DOI:<https://doi.org/10.1016/j.chb.2019.08.021>.
- [15] Schumacher, C. and Ifenthaler, D. 2018. The importance of students' motivational dispositions for designing learning analytics. *Journal of Computing in Higher Education*. 30, 3 (Dec. 2018), 599-619. DOI:<https://doi.org/10.1007/s12528-018-9188-y>.
- [16] Vieira, C., Parsons, P. and Byrd, V. 2018. Visual learning analytics of educational data: A systematic literature review and research agenda. *Computers & Education*. 122, (Jul. 2018), 119-135. DOI:<https://doi.org/10.1016/j.compedu.2018.03.018>.
- [17] Wijnia, L., Loyens, S.M.M., van Gog, T., Derous, E. and Schmidt, H.G. 2014. Is there a role for direct instruction in problem-based learning? Comparing student-constructed versus integrated model answers. *Learning and Instruction*. 34, (Dec. 2014), 22-31. DOI:<https://doi.org/10.1016/j.learninstruc.2014.07.006>.
- [18] Winne, P.H. 2020. Construct and consequential validity for learning analytics based on trace data. *Computers in Human Behavior*. 112, (Nov. 2020), 106457. DOI:<https://doi.org/10.1016/j.chb.2020.106457>.

CHUNK Learning: A Tool that Supports Personalized Education

Ralucca Gera
Naval Postgraduate School,
Monterey, CA, USA
rgera@nps.edu

D'Marie Bartolf
Naval Postgraduate School,
Monterey, CA, USA
dmarie.bartolf@nps.edu

Simona Tick
Naval Postgraduate School,
Monterey, CA, USA
sltick@nps.edu

Akrati Saxena
Eindhoven University of
Technology,
The Netherlands
a.saxena@tue.nl

ABSTRACT

The modern educational ecosystem is not one-size fits all. Students are accustomed to personalization in their everyday life and expect the same from education systems. Additionally, the COVID-19 pandemic placed us all in an acute teaching and learning laboratory experimentation which now creates expectations of self-paced learning and interactions with focused educational materials. Consequently, we examine how learning objectives can be achieved through a learning platform that offers content choices and multiple modalities of engagement to support self-paced learning and propose an approach to personalized education based on network science and data mining. This framework brings attention to learning experiences, rather than teaching experiences, by providing the learner engagement and content choices supported by a network of knowledge based on and driven by individual skills and goals. We further discuss the proposed prototype of such a learning platform, called CHUNK Learning. In this work, we present this tool, its benefits for students, challenges in personalized education, and future plans.

Keywords

CHUNK Learning Prototype, Personalized Education, Network of Knowledge, Network Data Mining.

1. INTRODUCTION

Education must meet the changing needs of a complex environment where learners are expected to contribute as creative problem-solvers. Education solutions must satisfy the specific educational needs of each learner while meeting the rapidly changing learning objectives for each degree or job in a resource-efficient way. Consequently, the educational ecosystem must also provide a flexible and rich cognitive

environment, supported by adaptable high-quality content for academic performance and complemented by effective learning analytics, as well as social networking for strong emotional support of students.

We identify the current shortcomings of learning platforms and propose solutions to improve personalized learning using data mining and network science techniques. At the content level, we focus on exploring the relevance of content by anchoring it to each learner's existing knowledge and how content connects to the skills of each learner. Additionally, we discuss how interconnecting people, content, goals, and skills support student learning outcomes and collaborative learning and what is the impact of these interactions on educational experiences.

Our goal is to facilitate a learning culture that cultivates curiosity and inquiry. While learners interact with different content to show the same proficiency, learning differentiation is based on branching off the main knowledge thread driven by each student's unique attributes, such as experience, existing skills, and learning goals. In this work, we propose a personalized education model driven by data science and network science, and present a solution, a prototype called CHUNK Learning [12]. We create this agile system by interconnecting the content, skills, and learners that creatively address learning theory while supporting an ecosystem that motivates students and improves learning outcomes. Our focus is on creating a network of knowledge, individualized education pathways, and a social network of users to improve personalized education.

2. OUR VISION AND STATE-OF-THE-ART

We present our vision for personalized education and the synthesis of state-of-the-art research in support of personalized education using network science and data mining.

2.1 The Design of a Network of Knowledge

Traditional education is linear, one chapter after another through a whole course or course after the course through a degree. An interconnected model of education brings an interconnected (non-linear) view of the knowledge, where a user can navigate through a network of knowledge built

R. Gera, D. Bartolf, S. Tick, and A. Saxena. CHUNK Learning: A tool that supports personalized education. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 743–747, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853115>

based on an ontology, pre-requisites, or dependency. Recent research identifies the benefit of interconnecting the knowledge for the learners, seeking models for a network of knowledge. For example, one way to model it is by “quantifying and analyzing the structure of students’ knowledge of a given discipline as a knowledge network of interconnected concepts”. Once this network is created, we can capture the learning pattern and retrieve the information of learners’ interactions over time [7, 17].

We propose a model that supports the interconnected world of 21st-century education, where subject matter experts co-create a network of knowledge as a curated collection of networked learning modules [1, 10]. A successful prototype was built and used since 2018, namely the Curated Heuristic Using a Network of Knowledge for Continuum of Learning (CHUNK Learning), i.e., a real-time, adaptive teaching-learning modular method for enhanced and personalized education.

2.2 The Education Pathways for Personalized Learning

Education is improving, but not as fast as our aspirations. Learners bring distinct backgrounds, learning style preferences, and different motivations for engaging with the content. This surfaces the need for personalized learning that assesses each learner’s gaps, skills, and prior experience before bringing in new information, thus preventing more gaps creation or repeat of lessons already learned.

It also requires that a variety of learning styles be available for learners when engaging with an instructor or the asynchronous content. Content and applications of the newly learned content must differ based on each learner’s experience and background; for example, learning and using a mathematical concept for an economist versus a mathematician is different. Content must have applications that are relevant to the learner’s background to anchor it to the learner’s experience since they will be applying it in the future. Besides this, some learners are directed to learn specific content for certification as needed by a job or degree, while some learners explore the network of knowledge for personal indulgence to up-skill their existing expertise. With the goal of providing a personalized experience to each student, choices of learning paths for each student need to be dynamically created. A method to create these learning paths can utilize network science and data mining techniques on the annotated network of content and users [7].

Cognitive flexibility theory discusses learning within complex and ill-structured knowledge domains through the inability of linear educational structures to support meaningful learning experiences [6]. In these environments, the network of knowledge demonstrates complex concepts’ interconnections and supports semantic memory. Additionally, the use of nodes and edges creates visual learning pathways that provide an avenue for supporting and tracking learners’ knowledge acquisition [17].

The use of network science to create a structural representation of learning paths supporting knowledge acquisition allows educators to determine if there is an optimal path. Networked education pathways provide the necessary envi-

ronment to examine the average length of paths between two nodes in the network and the relationship to students’ understanding of material [16]. This also presents an opportunity to examine the use of bridging concepts within a complex knowledge structure [17]. This will also provide a feedback mechanism to inform educators about how to improve educational experiences for learners, filling in gaps within the knowledge structure as needed in order to support desired learning outcomes. Once users have navigated even a portion of the network of knowledge, their experiences create a database of learning paths, which can be analyzed to understand the learning patterns based on gender, age, background, learner type (directed versus exploratory learner), or other attributes of the learners [13]. These findings can be used to further improve the personalized learning platform by suggesting what content should be added and designing a better recommendation system for learners.

2.3 Incorporating Online Social Networking in Learning Platforms

Online learning platforms complement or replace classroom learning for students; however, they usually lack students’ engagement. One way to support students’ engagement in such educational platforms is by incorporating social networking into the learning platform. Researchers have studied the benefits of networking the learners in learning and professional development [15]. Various existing platforms, such as Coursera [3], Moocs [4] provide the ability for students and instructors to communicate with each other. While this supports modern students’ learning, they lack social networking among students for content recommendation or live collaboration.

A learning platform can provide a service to maintain social connections for increasing users’ engagement in the following ways. First, users can follow or be friends with people they know or want to connect with for personal interest. Secondly, the platform should suggest new users to connect with based on similar backgrounds and same learning interests. Additionally, to incorporate group learning, the platform should suggest the teams either for instant collaboration or group projects based on users’ current learning paths. The recommendation system must be adaptive in recommending such teaming patterns based on the past performance of the students and their interests.

CHUNK Learning includes the vision for such a platform that is empowered by social networking to support meaningful learning timely and respectful of each learner’s time in a cost-effective manner [1]. Research on the CHUNK Learning platform considered the social network to recommend similar content to similar learners, as well as connecting similar learners based on their learning paths, background, and mentor-mentee type relationships [11].

3. A WORKING EXAMPLE: CHUNK LEARNING PLATFORM

We have designed a prototype of personalized education using the network of knowledge, learning pathways, and network of students, called CHUNK (Curated Heuristic Using a Network of Knowledge) Learning platform [1, 10, 12]. We now point out how CHUNK Learning fulfills some of the

CHUNK Explorer (right):

A curated collection of networked learning modules, organized to meet strategic objectives (academic or professional). Prerequisites directed by yellow arrows on the right.

Chunk (below): A self contained knowledge module.

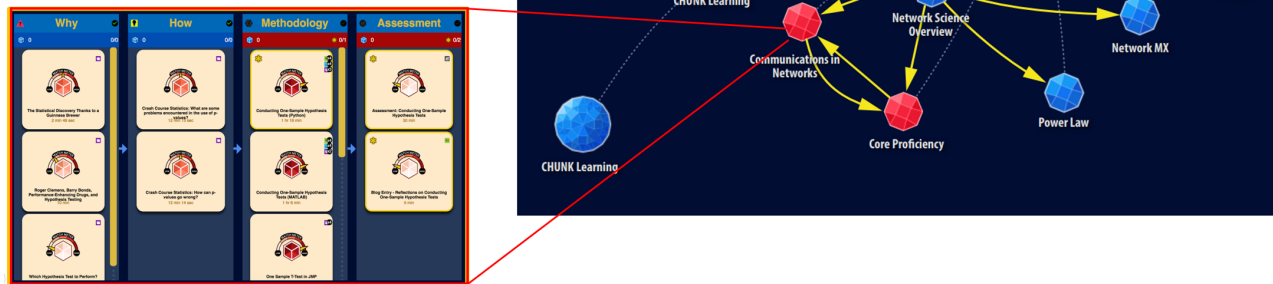


Figure 1: The network explorer and a chunk's content in CHUNK Learning [1]

identified needs for personalized education. Since the existing CHUNK Learning system is in its infancy stage, we will conclude this section with ideas that can extend the current version based on the discussion in Section 2.

The CHUNK Learning prototype offers several features identified in this work: (i) a personalized learning journey by using the information provided in the learner's profile to automatically recommend the most relevant-to-you content, (ii) content mapping that illustrates how a learner can progress through the network of knowledge by including the choices of learning paths through the network, and (iii) a framework to support social network of learners based on their profile and content they are engaging with at each time they log into CHUNK Learning. CHUNK Learning platform focuses on empowering students by ensuring that the learning journey is cohesive, flexible, and respectful of learners' time and interests. This personalized education is accomplished through the use of the following components.

- Content is chunked into intense, short, and focused educational modules (chunks) that are interconnected in the network of knowledge to provide (a) learner's choices and platform's recommendation to build personalized learning paths, (b) view of the learning path choices through the content, and (c) a big picture of cohesive learning.
- Choices of interchangeable and reusable content stimulate each learner's interest and provide relevance of each topic to each learner [10].
- Learners' are placed into a social network to support each other, such as pairing senior learners as mentors

for junior ones, based on their automatically updated profiles [14].

- Personalized content driven by implicit recommendations from a user's social network based on tie strength between learners [8].

Based on the learner's profile, CHUNK Learning optimizes both the content and the methodology delivery to meet the needs of each learner. It is intended to fit typical academic needs, much like curriculum mapping, while personalizing the content within the chunk of knowledge. Therefore, the content is organized hierarchically using Topics, Units, Chunks, and Chunklets to index content in a way that meets and improves the overall coherence of a course of study (more information on CHUNK Learning Wiki [2]).

Figure 1 shows the personalized adaptive learning framework by displaying a portion of the network of knowledge on the right and the personalized content of a chunk of knowledge on the bottom left of the figure. Within a chunk, the top row of content reveals the system's recommendations based on the learner's profile, and the other tiles are the choices of the ranked alternatives. The system displays content in a manner that can be viewed in a network format through the CHUNK Learning explorer at the topic, unit, and chunk levels. The explorer view at the topic-unit-chunk level is a concept mapping that follows a logical order to provide each learner a well-rounded and comprehensive educational experience, with choices for deeper dives and connections to other topics-unit-chunk. At the chunklet level, a learner can see the choices for all the building blocks needed to attain any specific academic competency

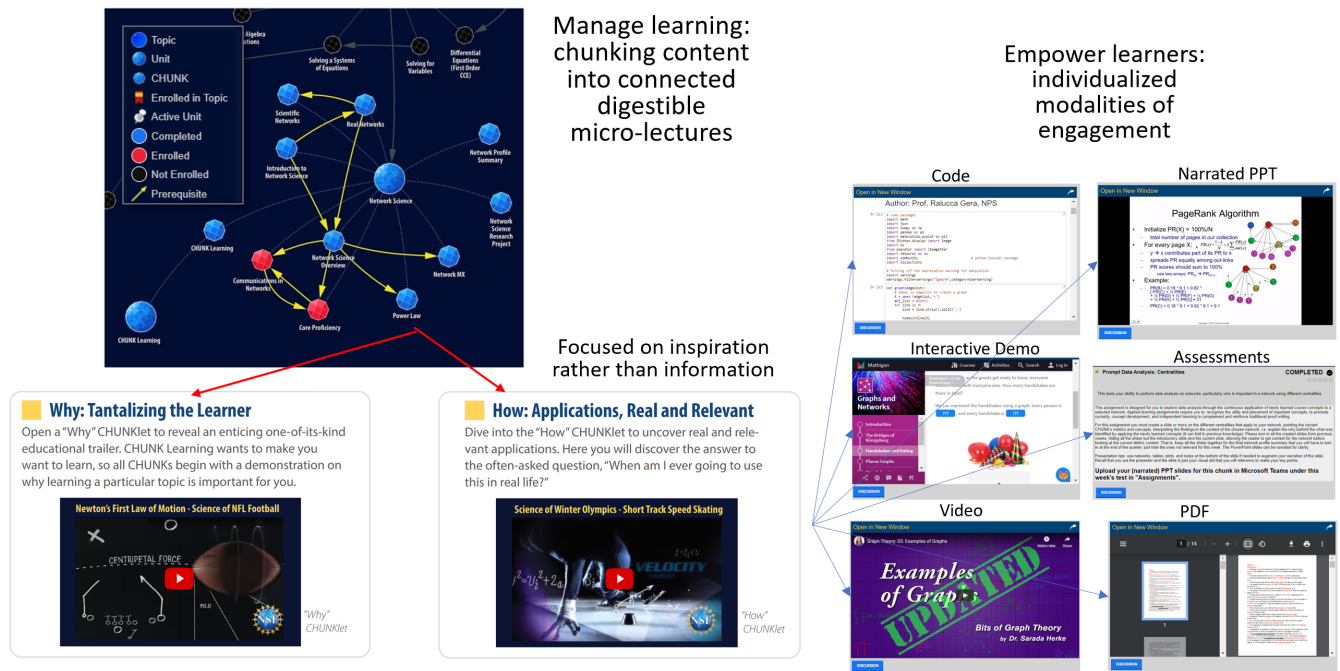


Figure 2: A view of personalized learning pathway in CHUNK Learning [1]

while being recommended the most relevant one based on that learner’s profile. Each column of a chunk provides the recommended chunklet at the top, followed by rank-ordered choices to complete that part of the chunk before heading to the assessment.

Figure 2 shows an example of personalization in CHUNK Learning achieved through (i) interchangeable chunklets that personalize the application of the content (on the left side of the figure) and (ii) personalized interchangeable activities for different styles of learning, such as video, PDF, code, demo, interactive activity (on the right side of the figure).

CHUNK Learning is a prototype used for hybrid teaching at our university. For these classes, students learn part of the content asynchronously using CHUNK Learning, complemented by synchronous practice and discussions. The assessment of students’ experience in these classes shows promising benefits in using CHUNK Learning for hybrid teaching. Besides the self-paced environment, students valued the mix of modalities of engagement with the content available on CHUNK Learning, supporting meaningful and engaging interactions with the class while enhancing their learning experience.

3.1 Extending the CHUNK Learning Platform

How can this prototype be further extended using network data mining? We see four primary areas of extension, and some progress has already been made in each of the areas: (i) the network of knowledge’s explorer view, (ii) the content within a chunk, (iii) the social network of users, and (iv) create a network-driven recommendation system and learning analytics. Below we expand on the existing work on each of these extensions.

CHUNK Learning’s explorer view of the Network of Knowledge in CHUNK Learning is static and identical for all users (except the color coding based on the enrolled and completed chunks). To focus and place the learner on a meaningful learning path, we propose a personalized display of the network of knowledge based on assessment and relevancy to the user’s learning objectives. Complementing the personalizing of the explorer view, we can personalize each chunk’s content. Currently, it is performed using the content’s tags compared to a learner profile’s tags, in addition to the ranking of the content based on the user’s rating of each chunklet. The two existing rankings are based on the quality of the content and relevance to the learner.

Currently, the recommender system updates the learning plan based on each learner’s activities (learned, viewed, tested), keyword searches, and content ratings [5, 9, 11]. We aim to update further the recommendation system using the social network of learners and their learning history.

4. CONCLUSION

In this paper, we introduced a prototype named CHUNK Learning to improve personalized education at scale using network data mining. The proposed prototype is based on the network of knowledge, education pathways, the network of learners, and the content recommendation system. We presented this tool, its existing features, and further plans. We also discussed some challenges to consider in creating and using such a personalized learning ecosystem. We aim to extend the tool further by including a social network of users to support personalized and collaborative learning. We share this work to promote conversations and research towards a new field of personalized education driven by network data mining.

5. REFERENCES

- [1] Chunk learning. <https://www.chunklearning.net>. Accessed: 2021-07-16.
- [2] Chunk learning wiki. <https://wiki.nps.edu/display/CHUNKL/Anatomy+of+a+CHUNK>. Accessed: 2021-07-16.
- [3] Coursera.org. <https://www.coursera.org/>. Accessed: 2021-07-16.
- [4] Mooc.org. <https://www.mooc.org/>. Accessed: 2021-07-16.
- [5] M. Andriulli, M. Smith, S. Smith, R. Gera, and M. L. Isenhour. Adaptive personalized network relationships in the chunk learning environment. In *The Fifth International Conference on Human and Social Analytics (HUSO'19)*, 2019.
- [6] S. R. Boger-Mehall. Cognitive flexibility theory: Implications for teaching and teacher education. In *Society for Information Technology & Teacher Education International Conference*, pages 991–993. Association for the Advancement of Computing in Education (AACE), 1996.
- [7] J.-D. Clevon. A multilayer network approach for real-time adaptive personalized learning. Technical report, Masters Thesis, Naval Postgraduate School, Monterey, CA, United States, 2018.
- [8] M. Critchley. A recommender model using social tie strength for the chunk learning system. Technical report, Masters Thesis, Naval Postgraduate School, Monterey, CA, United States, 2021.
- [9] D. O. Diaz, R. Gera, P. C. Keeley, M. T. Miller, and N. Leonardidis-Mena. A recommender model for the personalized adaptive chunk learning system. In *The Fifth International Conference on Human and Social Analytics (HUSO'19)*, 2019.
- [10] R. Gera, D. BartoIf, M. L. Isenhour, and S. Tick. Chunk: Curated heuristic using a network of knowledge. In *The Fifth International Conference on Human and Social Analytics (HUSO'19)*, 2019.
- [11] R. Gera, A. Gutzler, R. Hard, B. McDonough, and C. Sorenson. An adaptive education approach using the learners' social network. In *The Fifth International Conference on Human and Social Analytics (HUSO'19)*, 2019.
- [12] R. Gera, A. Saxena, D. BartoIf, and S. Tick. A network science perspective to personalized learning. *arXiv preprint arXiv:2111.01321*, 2021.
- [13] W. Lucero. Optimizing adaptive learning using statistical and network analysis within the chunk learning system. Technical report, Masters Thesis, Naval Postgraduate School, Monterey, CA, United States, 2020.
- [14] T. Reeder. Analysis of a similarity-based approach to positively affect mentor-mentee relationships among service members. Technical report, Masters Thesis, Naval Postgraduate School, Monterey, CA, United States, 2021.
- [15] A. Saxena, P. Saxena, H. Reddy, and R. Gera. A survey on studying the social networks of students. *arXiv preprint arXiv:1909.05079*, 2019.
- [16] C. S. Siew. Using network science to analyze concept maps of psychology undergraduates. *Applied Cognitive Psychology*, 33(4):662–668, 2019.
- [17] C. S. Siew. Applications of network science to education research: quantifying knowledge and the development of expertise through network analysis. *Education Sciences*, 10(4):101, 2020.

Do College Students Learn Math the Same Way as Middle School Students? On the Transferability of Findings on Within-Problem Supports in Intelligent Tutoring Systems

Michael Smalenberger
The University of North Carolina at Charlotte
msmalenb@uncc.edu

Kelly Smalenberger
Belmont Abbey College
kellysmalenberger@bac.edu

ABSTRACT

A widely used minimum criterion for intelligent tutoring system (ITS) status is that the system provides within-problem support. The ITS literature abounds with comparisons of supports, but investigations on the transferability of findings, particularly between education levels, are scant. There are, however, significant impediments to investigating the transferability of these findings. First, ITSs are used throughout K-16, and findings from past studies naturally serve as guideposts for subsequent ITS developments and research. If findings do not transfer, these guideposts may be illusory. Furthermore, learners change over time, and the efficacy of ITS supports that do not adapt to these changes may vary. This study conceptually replicates investigations conducted at the middle school level by assigning 330 college students in introductory mathematics courses homework that have the same within-problem support formats. The support formats are either text hints, text explanations, or video explanations. We compare the efficacy of these support formats within the college student sample and between college student and middle school student samples. Our findings at the college level indicate that on-demand within-problem explanations displayed as text rather than videos lead to higher next problem correctness and that both outperform text hints. Our results differ from those in the literature using middle school student samples and therefore buttress the assertion that studies investigating the transferability of findings between education levels are necessary.

Keywords

Educational Data Mining, ASSISTments, Within-Problem Supports, Replication, Intelligent Tutoring Systems

1. INTRODUCTION

A widely used criterion for intelligent tutoring system (ITS) status is that the system has an inner loop [1], i.e., provides within-problem support and not just end-of-problem feedback. The ITS literature abounds with comparisons of supports, including scaffolding versus hints [2], worked examples versus erroneous examples [3], and the supports that will be compared in this study, namely video hints versus text hints [4] and explanations versus hints [5]. Many ITS studies that evaluate the efficacy of these supports commonly involve a single segment of the student population,

e.g., middle school students. Investigations on the transferability of findings, particularly to college students, are scant.

There are, however, significant impediments to investigating the transferability of findings to other segments of the student population. First, ITSs are used throughout K-16, and findings from past studies naturally serve as guideposts for subsequent ITS developments and research. If findings do not transfer, these guideposts may be illusory. There is evidence to believe this is the case since [6] considered if the findings in [2] transferred to adult learners in a MOOC and found that the findings between hints and scaffolds did not transfer. Therefore, learners change over time, and the efficacy of ITS supports that do not adapt to these changes may vary.

For example, when describing what can be considered a good hint in an ITS, [7] in part states “Hints should abstractly (but succinctly) characterize the problem-solving knowledge (Anderson et al., 1995), by stating in general terms both the action to be taken, the conditions under which this particular action is appropriate, and the domain principle (e.g., geometry theorem) that justifies the action.” However, [8] asserted that young adolescents between the ages of 10-15 are in a transition period from concrete thinking to abstract thinking. Findings in [9] support this assertion and showed that middle school students benefited more from concepts with which they could directly relate, whereas college students benefited more from abstract concepts. This is relevant to within-problem supports while solving a mathematics problem. Hints which abstractly characterize the problem-solving knowledge, state in general terms the action to be taken and the conditions under which it is appropriate as [7] suggests may better serve college students who have the cognitive ability to assimilate and apply theorems and abstract definitions, whereas middle school students may be better served by an exposition through relatable or worked examples.

Similarly, a significant consideration in any instructional setting is Cognitive Load Theory (CLT) which stipulates that some difficulties in learning are due to unnecessary and extraneous working memory load. Here again, research has grappled with worked examples [10], including in comparison to tutored problem solving [11] and in ITSs [12, 13]. For example, the *redundancy principle* builds on the idea that learners utilize distinct information processing channels to internalize information, and hence stipulates that information should not be conveyed through multiple channels simultaneously to avoid depressing the intake from both channels and hampering learning. Research rooted in CLT at the middle school level compared the delivery medium of feedback messages within an ITS and stated that information is better internalized and learning gains are greater when feedback is presented as a video instead of text [4]. However, recent research suggests that executive function skills, which include monitoring and manipulating information in mind (working memory), suppressing distracting information and unwanted responses (inhibition), and flexible thinking (shifting), play a critical role in the development of

M. Smalenberger and K. Smalenberger. Do college students learn math the same way as middle school students? On the transferability of findings on within-problem supports in intelligent tutoring systems. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 748–752, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853117>

mathematics proficiency [14]. Although the literature is limited in terms of disentangling the neurodevelopmental aspects of specific cognitive processes in relation to their impact on arithmetic skill development, the research by [15] shows that with development there is a shift from prefrontal cortex-mediated information processing to more specialized mechanisms in the posterior parietal cortex. Freeing the prefrontal cortex from computational load and thus making available valuable processing resources for more complex problem solving and reasoning is a key factor in mathematical learning and skill acquisition [16]. According to [17], some of these processes, especially decision making and working memory, have protracted developmental timelines, but large changes can be detected in relatively small timeframes including from the beginning of middle school to the end of middle school [18, 19]. Therefore, when devising pedagogical supports, it is important to consider the differential effects on cognitive load depending on the student's attainment of neurodevelopmental milestones and subject matter fluency.

In this study, we break ground on investigating the transferability of findings from the middle school level to the college level regarding the efficacy of within-problem supports for mathematics ITSs. Specifically, this study conceptually replicates investigations conducted at the middle school level by assigning college students homework problems that have the same within-problem support formats. The support formats are either text hints, text explanations, or video explanations. We compare the efficacy of these support formats within the college student sample, and between college student and middle school student samples. For the within college sample investigation, we hypothesize that the findings from the replicated studies will transfer. For the between educational level investigation, we hypothesize that on average college student performance on the subsequent similar question will be similar given the support format (i.e., the relative frequency that the initial action is the correct solution will be equal).

Thus, we pose the following research questions:

1. Does the format of within-problem support impact learning outcomes for college students?
2. Are learning outcomes the same for college students and middle school students, given the within-problem support format?

2. BACKGROUND

For this study, we used ASSISTments which is a free web-based platform that allows teachers to choose from thousands of problems, each of which can provide immediate feedback and support. This support can be customized, e.g., it can be hint messages conveyed as either text or videos, explanations of step-by-step solutions, correctness feedback, etc. One facet of the ASSISTments ecosystem is E-TRIALS, which allows teachers and external researchers to create randomized control trials (RCTs) that run in ASSISTments to test the efficacy of supports.

The literature contains numerous RCTs using E-TRIALS which compare the efficacy of supports. To investigate the transferability of findings from the middle school level to the college level regarding the efficacy of supports, we will next briefly discuss the studies which we conceptually replicate.

2.1 Explanations Versus Hints

Cognitive scientists have investigated the role of worked examples and explanations in reducing cognitive load, and there have been several studies on their effectiveness [10, 12]. On the other hand, several studies in the literature have found evidence of the benefit of greater interaction in tutored problem-solving in ITSs [12, 13], including for less proficient students [11].

[5] investigated whether students in a classroom setting benefit more from interactive tutored problem solving than from worked

examples given as a feedback mechanism. For the experiment, researchers created nine problem sets, each consisting of four to five ASSISTments. The ASSISTments questions were taken from 6th Grade MCAS tests for Mathematics (2001 – 2007) focusing on the Patterns, Relations and Algebra section, which concentrates on populating a table from a relation, finding a missing value in a table, using fact families, determining equations for relations, substituting values into variables, interpreting relations from number patterns, and finding values from a graph.

In their study, 186 8th grade students from three middle schools participated. Over 80% of students who participated were from a school that had state test scores in the bottom 5% of that state. Results from this study show that there was a significant effect for the

condition with tutored problem solving receiving higher gain scores than worked examples (35% versus 13% average gain), $t(67) = 2.38, p = 0.02$.

2.2 Video Versus Text

Prior research has found that dynamic graphics are more effective than static graphics in mathematics realms [20], and videos and web-based homework support have grown increasingly popular in educational settings [21]. Some research suggests that video is not universally successful in promoting learning gains [22]. [4] assess the effects of feedback medium within a problem set derived from preexisting ASSISTments certified

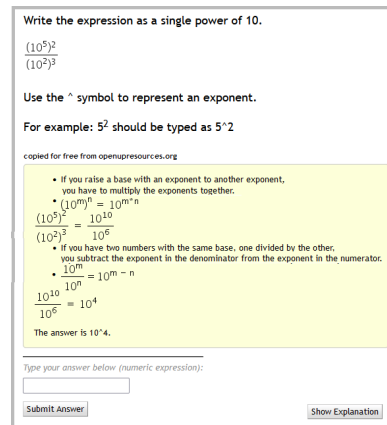


Figure 1. Example of Text Explanation

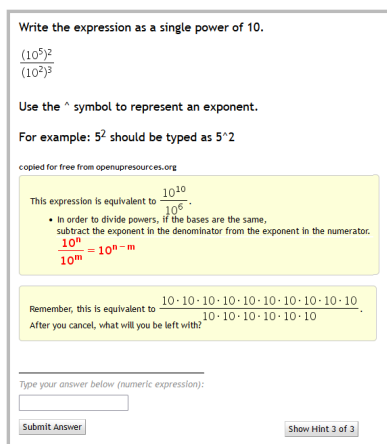


Figure 2. Example of Text Hints

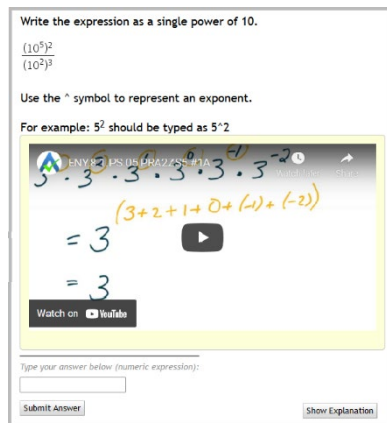


Figure 3. Example of Video Explanation

material. The structure of the problem set relied on three questions with text feedback and three isomorphic questions with video feedback and was presented to students in fixed question patterns to allow all students an equivalent opportunity to experience both feedback styles. Video content mirrored textual feedback to provide identical assistance through both mediums.

139 8th grade students from four classes comprised of four classes that spanned four suburban middle schools participated in this study. After exclusion criteria, 89 remained for analysis. The study's primary analysis assessed student performance on the second question as a function of the feedback medium they experienced after incorrectly answering the first question. As will be again addressed in the discussion section, learning outcomes were not enhanced for students who received video feedback rather than text feedback. There was also no significant difference in the overall number of feedback levels experienced by students.

3. METHODOLOGY

3.1 Procedure

330 college students in 6 sections of introductory mathematics courses at two institutions participated in this study. Demographic information of this group is as follows: 9% transfer students, 56% female, 44% male, 52% White, 19% Black, 16% Hispanic, 7 % Asian, 4% two or more races, 1% International, and 1% Unknown.

Students were assigned up to 5 problems sets (PS) as homework during the last three weeks of the Fall 2021 semester in a manner that was consistent with regular educational practices. Each problem set covered one topic, namely rational exponents (PS 1), multiplying and dividing monomials (PS 2), scientific notation (PS 3), writing linear equations given ordered pairs (PS 4), and systems of linear equations (PS 5). A problem set was only assigned to a class if it fell within the course's curriculum.

Students were free to work at their own pace and were not required to complete the assignments in one sitting. Log data was recorded for each student's actions, including first action, correctness, response time, attempts, hint requests, and feedback type, and this data was received from The ASSISTments Foundation.

3.2 Design

The five problem sets were made using preexisting ASSISTments certified material. Problem sets 1-3 consisted of 30 problems each, and problem sets 4 and 5 consisted of 20 problems each. Problems had between 1 and 9 parts, for a combined total of 196 parts in all problem sets. Each part had exactly one input field which required a free-response answer usually consisting of a number or mathematical expression. Questions within a problem set were assigned in random order to eliminate potential ordering effects.

Students could attempt any part an unlimited number of times until they entered the correct answer. Each part had on-demand within-problem supports consisting entirely of either text hints, text explanations, or video explanations. Students were aware that there were different kinds of support types but did not know which types of support any particular question had. If a student requested within-problem support, we relied on ASSISTments to randomly assign the student one of the available supports.

Our analysis assesses student performance on the question part immediately after the question part for which support was requested, which we will refer to as the next question and previous question, respectively. As such, one of our exclusion criteria is that any problem for which a student did not complete the next question is omitted. We assess student performance based on the next problem's correctness which we refer to as learning outcomes. To

that end, we consider the next question to be incorrect if a student's first action is to either submit a wrong solution or request support.

4. RESULTS

4.1 Next Question Performance Analysis Within College Student Sample

To address our first research question, we assessed the next question performance of students who received feedback on the previous question. In total, 40126 solution steps were completed by students included in our analysis, and on 4820 solution steps (4820/40126 = 0.120) supports were provided, as Table 1 shows.

Learning outcomes were more enhanced for students who received text explanations ($\hat{p} = 0.493$, $SD = 0.009$) rather than video explanations ($\hat{p} = 0.390$, $SD = 0.018$), and the difference in proportions of 0.103 was statistically significant ($p \ll 0.01$). This indicates that students answered the next question correctly over 26 percent more often ($0.493/0.390 = 1.264$) after receiving on-demand within-problem text explanations than video explanations.

Table 1. Summary of Next Question Performance Analysis

Support Format 1	Support Format 2	Difference in Proportions	95% Confidence Interval for Difference in Proportions
Text Explanation (N = 3161)	Video Explanation (N = 757)		
0.493 (0.009)	0.390 (0.018)	0.103*	[0.064, 0.142]
Text Explanation (N = 3161)	Text Hints (N = 902)		
0.493 (0.009)	0.135 (0.011)	0.358*	[0.329, 0.386]
Video Explanation (N = 757)	Text Hints (N = 902)		
0.390 (0.018)	0.135 (0.011)	0.254*	[0.213, 0.296]

Note: Performance metrics are depicted as mean (standard deviation). * $p \ll 0.01$.

Learning outcomes were enhanced for students who received text explanations rather than text hints ($\hat{p} = 0.135$, $SD = 0.011$). The difference in proportions of 0.358 was statistically significant ($p \ll 0.01$) indicating that students were over 3.5 times more likely ($0.493/0.135 = 3.652$) to answer the next question correctly after receiving ext explanations than text hints.

Learning outcomes were enhanced for students who received video explanations rather than text hints, and the difference in proportions of 0.254 was statistically significant ($p \ll 0.01$). This indicates that students were nearly 3 times more likely ($0.390/0.135 = 2.889$) to answer the next question correctly after receiving on-demand within-problem video explanations than text hints.

5. DISCUSSION

5.1 Comparison of Next Question Performance Between Education Levels

To address our second research question, we compare the enhancements of learning outcomes for middle school students to those

from our college student sample using the combinations of support formats in [4] and [5].

Before drawing comparisons to our results, it should first be noted that the statistical analysis in [4] has apparent flaws based on the data provided in that paper, and these flaws affect the conclusion of that study. After correcting those analysis errors using the data the authors provided in [4], the correct conclusion of [4] should be that the modality of the support in that study did not have a statistically significant effect on learning outcomes. Nevertheless, the corrected results of [4] are only partially reflected in our results. That is, college students were nearly 3 times more likely to answer the next question correctly after receiving video explanations rather than text hints, but over 26 percent more likely to answer the next question correctly after receiving text explanations rather than video explanations. As previously indicated, both results are also statistically significant.

Another important difference to note is the magnitudes of the efficacies of the different forms of support. At the middle school level, [4] found that students who received video support answered the next question correctly 76% of the time on average, whereas in our study students who received video support answered the next question correctly only 39% of the time on average, with a 95% confidence interval of [0.354, 0.426]. Furthermore, [4] found that middle school students who received text hints answered the next question correctly 52% of the time on average, whereas in our study students who received text hints answered the next question correctly only 13.5% of the time on average, with a 95% confidence interval of [0.113, 0.157]. Therefore, the results in [4] fall well outside of our expected range.

[5] found that learning outcomes were greater for middle school students who received tutored problem solving rather than worked examples and that this difference was statistically significant. This is not reflected in our results since college students were over 3.5 times more likely to answer the next question correctly after receiving on-demand within-problem text explanations than text hints. As previously indicated, the results from our sample are also statistically significant.

Since the results in [5] are presented as average learning gains from a pre- to a post-test, and our results are given as next problem correctness, we cannot draw similar point estimate comparisons to that study.

5.2 Contributions, Limitations, and Future Research

We make a significant contribution by breaking ground on investigating the transferability of findings from the middle school level to the college level regarding the efficacy of on-demand within-problem supports for mathematics ITSs. The ITS literature abounds with investigations using on-demand within-problem supports, particularly for middle school and high school students. Some studies have compared the efficacy of these supports and have led to what has been coined “Best So Far” supports – those supports which have led to the highest measures of learning outcomes for comparable studies [23, 24]. Our conceptual replication of these studies with college students did not support the same hierarchy of supports nor magnitude of efficacy at the college level.

This beckons the question, “Best So Far” for *whom*? To that end, our study identifies a rich area of research since there are abundant ITS studies that often include only one segment of the student population. While it is understandably hard to include students from several education levels in a study, particularly for mathematics ITSs, our results buttress the assertion that studies investigating the transferability of findings between education levels are

necessary. Our study also lends credence to the importance of considering who is or is not included in a study, and disaggregating results when possible to ensure inclusion, diversity, equity, and accessibility in research and practice.

Another rich area of research is to data-mine the results from such transferability studies to supplement the limited literature on neurodevelopmental aspects of specific cognitive processes in relation to their impact on learning and best practices for instruction.

Future iterations in this line of research should consider a closer replication of the literature. While our study investigated the modality effects of on-demand within-problem supports, we compared video *explanations* to text *explanations*, while [4] compared video *hints* to text *hints*. Similarly, while our study compared the impact of interactive forms of tutored problem solving to static forms of support, our study used *text explanations* and text hints, whereas [5] used *worked examples* and text hints.

Future research should also consider that while in this study problem sets were only assigned to a class if they fell within the course’s curriculum, a potential concern is that likely exposure to these topics in prior courses may lead to a college student being more familiar with the topics in the problem sets than middle school students, resulting in a confound that could influence results. Possible indicators of this would be fewer support requests from college students, or higher measures of next problem correctness. Our results indicate the opposite. As was previously shown, college students requested support on 12 percent of solution steps ($4820/40126 = 0.120$), but in [4] middle school students requested support on approximately 8 percent of solution steps ($60/720 = 0.083$). Furthermore, as was also previously shown, next problem correctness was consistently lower for college students than for middle school students when comparing similar supports.

On the other hand, since the assignment topics in the original studies differ from each other and from those used in this study, it can be argued that the difficulty of these topics may also differ, again leading to a confound. This too may be indicated by different levels of support requests or different levels of next problem correctness. Based on the just presented discussion, one conclusion which could be drawn from our results may indicate that the topics covered in the problem set by the college students were more difficult than those completed by middle school students. Another conclusion could be that the problems in assignments completed by college students were equally difficult to those completed by middle school students and that the overall relative mathematics skill levels of the two groups differed.

Disentangling these results further is not possible using the results of our study or the available information from the replicated studies. While challenging to implement, something that could mitigate this ambiguity is for future research to include a measure comparing mathematics abilities across education levels, e.g., a pretest, and assign identical problem sets to all study participants across education levels. Furthermore, by including a posttest, future research should also further analyze other learning outcomes, such as that of [5].

Finally, an exclusion criterium in this study is that any problem for which the student did not complete the next question is omitted, and therefore attrition may not have been entirely random. While we do not believe this had any substantial impact on our results, future research should definitively ensure this is true by establishing a more controlled experimental environment.

6. ACKNOWLEDGEMENTS

We thank The ASSISTments Foundation. As always, K.H.S., J.M.S., E.M.S., and W.J.S. thank you and I.I. y.

7. REFERENCES

- [1] VanLehn, K., 2006. The behavior of tutoring systems. *International journal of artificial intelligence in education*, 16(3), pp.227-265.
- [2] Razzaq, L. and Heffernan, N.T., 2006, June. Scaffolding vs. hints in the Assistment System. In *International Conference on Intelligent Tutoring Systems* (pp. 635-644). Springer, Berlin, Heidelberg.
- [3] McLaren, B.M., van Gog, T., Ganoë, C., Karabinos, M. and Yaron, D., 2016. The efficiency of worked examples compared to erroneous examples, tutored problem solving, and problem solving in computer-based learning environments. *Computers in Human Behavior*, 55, pp.87-99.
- [4] Ostrow, K. and Heffernan, N., 2014, July. Testing the multimedia principle in the real world: a comparison of video vs. Text feedback in authentic middle school math assignments. In *Educational Data Mining 2014*.
- [5] Shrestha, P., Maharjan, A., Wei, X., Razzaq, L., Heffernan, N.T. and Heffernan, C., 2009. Are worked examples an effective feedback mechanism during problem solving. In *Proceedings of the 31st annual conference of the cognitive science society* (pp. 1294-1299).
- [6] Zhou, Y., Andres-Bray, J.M., Hutt, S., Ostrow, K., Baker, R. S., 2021. A Comparison of Hints vs. Scaffolding in a MOOC with Adult Learners. In Roll, I., McNamara, D., Sosnovsky, S., Luckin, R., Dimitrova, V. (eds) *Artificial Intelligence in Education. AIED 2021. Lecture Notes in Computer Science* (), vol 12749. Springer, Cham. https://doi.org/10.1007/978-3-030-78270-2_76
- [7] Aleven, V., Roll, I., McLaren, B.M. and Koedinger, K.R., 2016. Help helps, but only so much: Research on help seeking with intelligent tutoring systems. *International Journal of Artificial Intelligence in Education*, 26(1), pp.205-223.
- [8] National Middle School Association and Believe, T.W., 1995. *Developmentally Responsive Middle Level Schools. A Position Paper of the National Middle School Association. NMSA. Columbus, Ohio: NMSA.*
- [9] Song, H.D., Grabowski, B.L., Koszalka, T.A. and Harkness, W.L., 2006. Patterns of instructional-design factors prompting reflective thinking in middle-school and college level problem-based learning environments. *Instructional Science*, 34(1), pp.63-87.
- [10] Chi, M.T., Siler, S.A., Jeong, H., Yamauchi, T. and Hausmann, R.G., 2001. Learning from human tutoring. *Cognitive science*, 25(4), pp.471-533.
- [11] Razzaq, L., Heffernan, N.T. and Lindeman, R.W., 2007. What level of tutor feedback is best. In *Proceedings of the 13th Conference on Artificial Intelligence in Education, IOS Press*.
- [12] Graesser, A.C., Moreno, K., Marineau, J., Adcock, A., Olney, A., Person, N. and Tutoring Research Group, 2003. AutoTutor improves deep learning of computer literacy: Is it the dialog or the talking head. In *Proceedings of artificial intelligence in education* (Vol. 4754).
- [13] Graesser, A., Jackson, G.T., Jordan, P., Olney, A., Rose, C.P. and vanLehn, K., 2005. When is reading just as effective as one-on-one interactive human tutoring?. In *Proceedings of the Annual Meeting of the Cognitive Science Society* (Vol. 27, No. 27).
- [14] Cragg, L. and Gilmore, C., 2014. Skills underlying mathematics: The role of executive function in the development of mathematics proficiency. *Trends in neuroscience and education*, 3(2), pp.63-68.
- [15] Menon, V., 2010. Developmental cognitive neuroscience of arithmetic: implications for learning and education. *Zdm*, 42(6), pp.515-525.
- [16] Van Merriënboer, J.J. and Sweller, J., 2005. Cognitive load theory and complex learning: Recent developments and future directions. *Educational psychology review*, 17(2), pp.147-177.
- [17] Kwon, H., Reiss, A.L. and Menon, V., 2002. Neural basis of protracted developmental changes in visuo-spatial working memory. *Proceedings of the National Academy of Sciences*, 99(20), pp.13336-13341.
- [18] Viterbori, P., Usai, M.C., Traverso, L. and De Franchis, V., 2015. How preschool executive functioning predicts several aspects of math achievement in Grades 1 and 3: A longitudinal study. *Journal of Experimental Child Psychology*, 140, pp.38-55.
- [19] Holmes, J. and Adams, J.W., 2006. Working memory and children's mathematical skills: Implications for mathematical development and mathematics curricula. *Educational Psychology*, 26(3), pp.339-366.
- [20] Mayer, R.E. (Ed). 2005. *The Cambridge handbook of multimedia learning*. New York: Cambridge University Press
- [21] Kelly, K., Heffernan, N., D'Mello, S., Namias, J., & Strain, A. 2013. Adding teacher-created motivational video to an ITS. In *Proceedings of the Florida Artificial Intelligence Research Society Conference (FLAIRS 2013)*. 503-508.
- [22] Pane, J.F. 1994. *Assessment of the ACSE science learning environment and the impact of movies and simulations*. Carnegie Mellon University, School of Computer Science Technical Report CMU-CS-94-162, Pittsburgh, PA.
- [23] Patikorn, T. and Heffernan, N.T., 2020, August. Effectiveness of crowd-sourcing on-demand assistance from teachers in online learning platforms. In *Proceedings of the Seventh ACM Conference on Learning@ Scale* (pp. 115-124).
- [24] Prihar, E., Patikorn, T., Botelho, A., Sales, A. and Heffernan, N., 2021, June. Toward Personalizing Students' Education with Crowdsourced Tutoring. In *Proceedings of the Eighth ACM Conference on Learning@ Scale* (pp. 37-45).

Comparison of Learning Behaviors on an e-Book System in 2019 Onsite and 2020 Online Courses

Hiroaki Kawashima
Graduate School of Information Science
University of Hyogo, Kobe, Japan
kawashima@gsis.u-hyogo.ac.jp

ABSTRACT

In this study, we compare e-Book log data from onsite face-to-face courses in 2019 and synchronized online courses in 2020 to elucidate the difference in students' learning behaviors. We focus on short periods before and after class time by considering locational and temporal constraints in onsite courses, i.e., students should be physically presented in a classroom and leave the room after each class. As online courses are free from such constraints, we show that learners' behavior in 2020 has characteristic patterns before and after class time and that the activity of students in those periods relates to their final grades.

Keywords

Learning analytics, onsite vs. online, e-book system

1. INTRODUCTION

The learning environment has changed drastically due to the COVID-19 pandemic. In particular, online classes have been widely used when the number of infections is large. There are several styles of online classes, such as synchronized and asynchronized [3]. While asynchronized courses make students watch prerecorded videos in an on-demand fashion, synchronized online courses often require students to connect to a video-conferencing system and attend the class online from their homes at a specified class time. In addition, the media transmitted by the video-conferencing system is not limited to video of a lecturer captured by a camera, but a variety of other styles are used; for example, if class materials (e.g., slides) are shared in another way, it is enough to transmit only audio of lecturers' speech.

How have learning styles changed with the introduction of such synchronized online classes? This study aims to clarify changes in learning behaviors and their relationship to final grades (e.g., A, B, and C) by analyzing browsing log data from 2019 onsite face-to-face classes and 2020 synchronized online classes in university lectures where the e-Book

system has been introduced. We focus on the fact that the face-to-face format has locational and temporal constraints that require students to go to the classroom at a specific time and leave the room just after class, whereas the online format frees students from such constraints. From these considerations, in this study, we particularly analyze students' activity in short periods just before and after class time to understand the difference in learning styles between 2019 and 2020. The contributions of this study are twofold:

1. We show that onsite face-to-face and online teaching formats cause different patterns of students' activity during short periods of time (e.g., 15 minutes) before and after class time.
2. We also show that the activity patterns before and after class time relate to students' final grades.

2. RELATED WORK

Clickstreams from e-Book systems are often used in the field of learning analytics to provide detailed analysis of how students are viewing course materials [16, 14]. Such log data is used for a wide range of applications, including feedback to instructors during classes [13], grade prediction [10, 2, 8], and summarization of teaching materials [15]. For final grade or score prediction, methods for identifying important features [5] and predicting final grades using various machine learning methods have been proposed [10, 2, 8]. This can also be used for early dropout prediction [2].

Comparisons between onsite (face-to-face) and online classes have been studied in the education of various fields. Aggarwal et al. [1] conducted a randomized study to test the difference in learning effects between onsite and online courses on Biostatistics and Research Ethics and found no significant differences. Jones and Long [6] compared onsite and online students in the same mathematics course from 2005 to 2011 and showed that onsite students performed better for overall ten semesters but were comparable when limited to the last seven semesters. Paul and Jefferson [12] conducted a comparative analysis in an environmental science course from 2009 to 2016 and reported no difference in academic performance between onsite and online formats.

There are some later studies conducted after the onset of the COVID-19 pandemic. For example, Yang et al. [17] examined the viewing time and completion rates of online dental education courses and found that the completion rate

H. Kawashima. Comparison of learning behaviors on an e-book system in 2019 onsite and 2020 online courses. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 753–757, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853147>

Table 1: Basic information of the courses in the dataset.

Course ID	Weeks	# of students	Class time			
A-2019	8	50	8:40-12:00 (Mon.)			
A-2020	7	62	8:40-12:00 (Mon.)			
B-2019	8	164	B-2020	7	93	14:50-16:20 (Tue.)
B-2020	7	93	14:50-16:20 (Tue.)			

was related to the time the content was first visited (within 60 minutes before, further before, or after the start of the class). Amin et al. [3] analyzed the end-of-course surveys in electronic circuit courses from 2007 to 2021 and reported that online classes provided as good as or better learning experiences than onsite classes.

Many of the existing studies mentioned above mainly compared onsite and online courses based on grades/questionnaires or analyzed students’ learning behaviors such as viewing duration of contents and first accessed time. On the other hand, this study aims at finding a new perspective to compare onsite and online students’ behaviors by defining the amount of activity using detailed operation logs of an e-Book system in short periods of time before and after classes.

3. LECTURE COURSES AND DATASET

3.1 Courses and Teaching Formats

The dataset used in this study was open for the Data Challenge at the International Conference on Learning Analytics & Knowledge 2022 [9, 4, 7] and provided on a request basis. It consists of four courses whose course IDs are A-2019, A-2020, B-2019, and B-2020, where "A" and "B" denote the subjects of the courses, and the numbers 2019 and 2020 denote the offered year. For simplicity, we denote the pair of courses X-2019 and X-2020 as "course X" in the following. Each of course A and course B covers the same topics for both years (e.g., B-2019 and B-2020 dealt with the same topics).

In 2019, the teaching format was onsite face-to-face, and each student operated the e-Book system in a classroom using his or her own laptop computer while listening to a slide presentation by a lecturer on the podium. In 2020, on the other hand, the teaching format was synchronized online. Lecturers gave audio-only lectures during each class time, and students used their own PCs to operate the e-Book system at home or from other locations.

3.2 Class Weeks and Materials

While the topics are essentially the same within each of courses A and B, the number of weeks and materials differs over the two years. First, both courses A and B were offered for eight weeks in 2019 and seven weeks in 2020. In course A, the content of Weeks 5 and 6 in 2019 was taught in Week 5 in 2020, and in course B, the content of Weeks 2 and 3 in 2019 was taught in Week 2 in 2020. In addition, the course materials were updated in 2020 to some extent. In course A-2019, a summarized version of the materials was also distributed on the e-Book system. Basic information about each course is shown in Table 1.

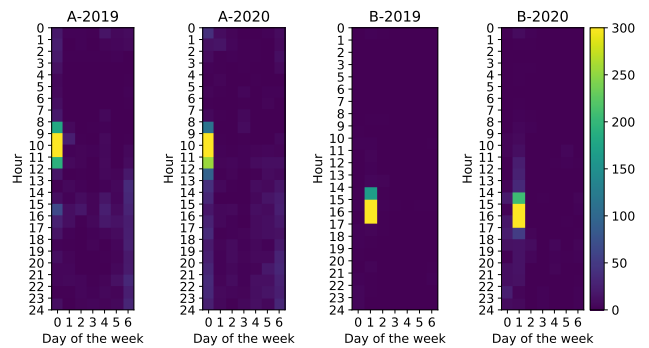


Figure 1: Visualization of activity in each day of the week (0: Monday, 6: Sunday). Active periods correspond to the class time of each course. See also Table 1.

3.3 e-Book Log Data

The data used in this study are students’ operation logs obtained through an e-Book system [9, 4]. Students browse provided class materials (e.g., slides) by performing, for example, page forwarding and marking operations. Each line of the log data records what operation is performed by which student on which page of which course material, along with a timestamp consisting of date, hour, minute, and second. Types of operations include page transitions, adding notes and markers, and bookmarking. However, as described later, this paper focuses on the number of operations, which we also refer to as operation count or frequency, as the amount of activity. All data are anonymized, and timestamps are slightly blurred on a scale of seconds while maintaining order from the original.

Each student’s final grade is recorded as A, B, C, D, or F in every course, where A is the highest grade, and F is the lowest grade. The records are also included in the dataset.

4. ANALYSIS

4.1 Visualization of Activity by Time of Day

To examine the overall trend of students’ activity for each course, we first visualize the total activity at each hour of each day of the week. We here define "activity" as the frequency of operations without distinguishing operation types, where page transition operations (moving to the previous or next page) comprise the majority of activities.

Figure 1 shows the visualization of total activity in each hour. The color in each time slot shows the total frequency of operations through the course period. Note that the frequency is normalized by dividing by the number of students in the course. As for the colormap range, we set 300 as the highest value to be covered (colored by yellow). The figure shows that the time slots with the highest activity correspond to the class time of each course.

When the value on a particular grid is significantly high, the other grids are buried in this kind of heatmap visualization. Therefore, we visualize the same data with different colormap that ranges from 0 to 40 in Fig 2, i.e., each time slot with yellow color has an average of 40 operations or more per student. We can see in this figure that the amount

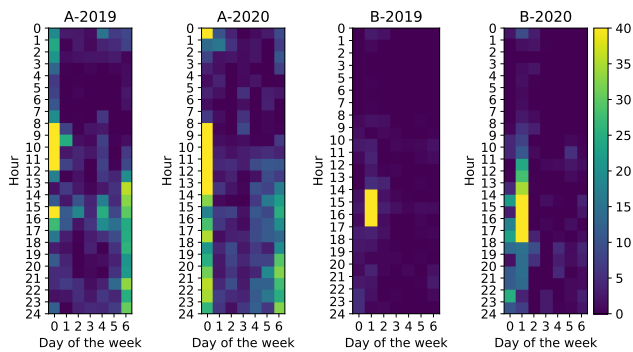


Figure 2: Visualization of activity in each day of the week (0: Monday, 6: Sunday) with colormap ranges from 0 to 40.

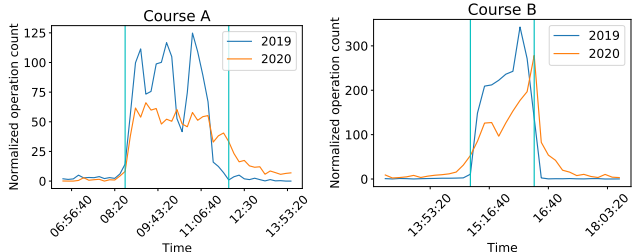


Figure 3: Activity patterns around the class time in courses A and B. Vertical lines show the start and end of class.

of activity on Sundays is relatively high in both 2019 and 2020 for course A. This may be due to the deadline setting of assignments. In addition, both course A and course B have more activity outside of class time on the day of the class in 2020 compared to 2019.

To detail the activity patterns around the class time, we aggregated activity with 10-minute intervals during the class time with a margin of two hours before and after the class, which is shown in Fig. 3. Compared to 2019, the patterns in 2020 are characterized by before and after the class time. In A-2020 and B-2020, activity continues after the class for a certain period, which we refer to as “post-class activity.” In addition, in course B-2020, activity increases before the class starts, namely “pre-class activity” exists. On the other hand, A-2020 has no significant pre-class activity, which may be because course A was scheduled in the early morning while course B was taught in the afternoon.

4.2 Relationship to Final Grades

To further investigate the pre-/post-class activity, we divided students into groups based on their final grades and examined the differences in the amount of pre-/post-class activity. Figure 4 shows the average pre-/post-class activity in each grade level. Specifically, we first calculated the pre-/post-class activity of each student based on the total operation frequency during a specific period (i.e., 15min before or after class) on all class days. We then calculated the average students’ pre-/post-class activity in each grade level. Error bars indicate 90% confidence intervals using t -distributions. We can find that, in 2020, the amount of post-class activity tends to be higher the better the final grade.

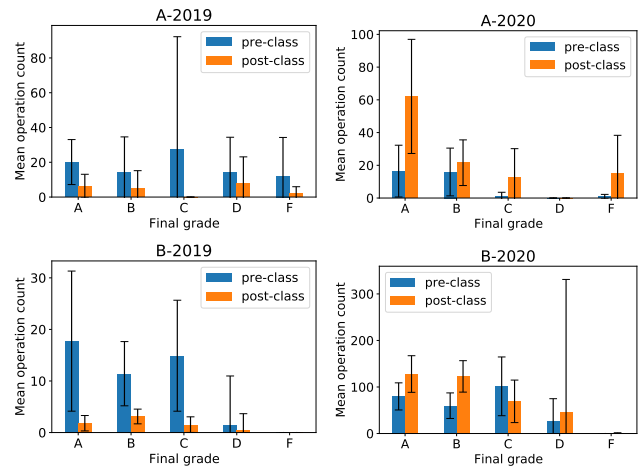


Figure 4: Average pre-/post-class activity (the number of operations) in each grade level. Blue bars: pre-class activity; orange bars: post-class activity. Error bars show 90% confidence intervals.

Table 2: Results of the Kruskal-Wallis tests for post-class activity. The rightmost column shows the sample size of each grade.

Course ID	p -value	n (A, B, C, D, F)
A-2019	0.7023	(24, 6, 4, 6, 10)
A-2020	0.0067	(22, 24, 6, 3, 7)
B-2019	0.4088	(26, 104, 30, 2, 2)
B-2020	0.0078	(37, 38, 12, 2, 4)

For each course, we conducted a statistical test for the difference in the amount of post-class activity among grade levels. The results (p -values) of the Kruskal-Wallis tests are shown in Table 2, indicating significant differences among grade levels in 2020 for both courses A and B. While the computation of the confidence intervals in Fig. 4 assumes that each population is normally distributed, the actual activity distribution tends to be biased toward lower values and often contains outliers. We, therefore, used nonparametric tests here and in the following.

In order to examine in detail which grade groups have the gap of activity, we conducted the Mann-Whitney U-test separately for divided two-group pairs: grade group {A} vs. {B, C, D, F}, {A, B} vs. {C, D, F}, and {A, B, C} vs. {D, F}. The reason for grouping the grade levels in this way is that the criteria for grading may differ from course to course; for example, two grade levels in one course may correspond to a single grade level in another course. We did not include only grade F as a group because of the small sample size.

The results are shown in Table 3. We can find from the table that some grade groups have significant gaps in the post-class activity. For example, in course A-2020, there are significant differences between grade group {A} and {B, C, D, F}, which corresponds to the gap in orange-bar heights in Fig. 4. In course B-2020, between {A, B} and {C, D, F}.

Table 3: Results (p -values) of the Mann-Whitney U-tests for the post-class activity. Vertical lines in the column names show how the grade levels were divided into two groups.

Course ID	A BCDF	AB CDF	ABC DF
A-2019	0.6221	0.2676	0.5449
A-2020	0.0005	0.0139	0.0317
B-2019	0.8662	0.0786	0.3716
B-2020	0.4232	0.0020	0.0023

5. DISCUSSION

The results of the analyses in the previous section suggest that the change of teaching format from onsite face-to-face classes in 2019 to synchronized online classes in 2020 frees students from constraints of class time and locations. As a result, some students continue to examine class materials even after class time, observed as the post-class activity. Such students' behavior on a relatively short time scale (e.g., 15 minutes after class) may provide important clues for understanding students' learning in online environments.

The pre-/post-class activity might be an indicator of students' self-regulation [11] in some sense. However, it should be examined with further study combining other clues, such as learning behaviors (e.g., the frequency of reviewing class materials), learning strategies, and motivations.

Some limitations exist in our analyses in the previous section. One is the effect of class attendance. As we computed the amount of activity from the total frequency of operations in a specific time period, it also contains information on attendance or absences. For example, if a student is absent from a class, post-class activity may also become close to zero on that day. While we decided to include such effects in the present study, in order to investigate pre-/post-class activity conditioned on "attended" students, we need further analysis by estimating whether each student attended or not on each class day.

Another limitation is that the objective of this study is not on the utilization of features, e.g., improvement of prediction accuracy of grade levels. Therefore, it is unclear whether features extracted from the pre-/post-class activity are useful, and other activities (e.g., in-class activity) can be more critical, for example, for final-grade prediction. To directly compare the importance of pre-class, post-class, and in-class activity, we trained gradient boosting classifiers, widely used for various data challenges including educational data.

Figure 5 shows the feature importance obtained through the model training. Here, we used the LightGBM implementation. Note that our objective here is not the evaluation of prediction accuracy itself but a brief comparison of feature importances. We, therefore, trained a model from all data in each course. We also constrained model complexity to a certain degree with the hyperparameters (the maximum tree depth: 3, the minimum number of data in a leaf: 10, and the number of iterations: 20) to avoid too much overfitting. As for the in-class activity feature, we computed operation frequency during class time in the same manner as pre-/post-class activation.

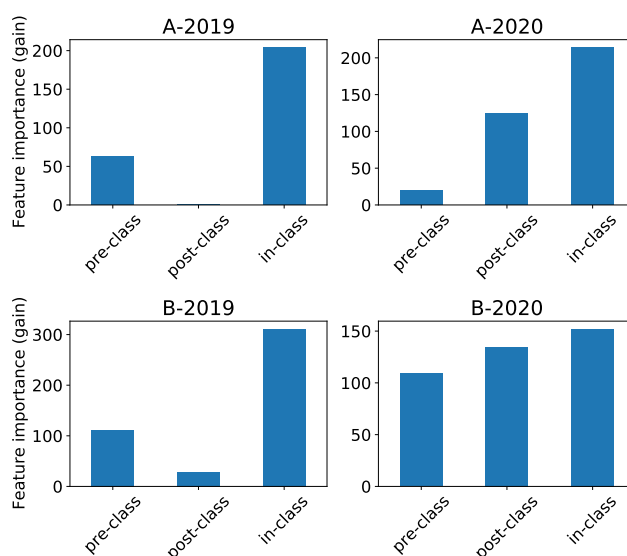


Figure 5: Feature importances computed by the gain of gradient boosting models.

From the figure, in-class activity seems to be the most crucial feature for the grade prediction task throughout 2019 and 2020. Meanwhile, it is worth noting that the importance of pre-/post-class activity substantially increased in 2020 compared to 2019.

While appropriate evaluation of the grade-prediction task requires appropriate cross-validation with a larger amount of data, the results of this study show some potential to design and combine various detailed features in out-class time periods in the context of online courses.

6. CONCLUSION

This study analyzed student behavior in 2019 and 2020 before and during the COVID-19 pandemic, based on operation logs obtained through an e-Book system, for two courses whose teaching format changed from face-to-face to online in the two years. Because online classes do not have the restriction of leaving the lecture room after class, activity after class, which was not seen in 2019, was often observed in 2020. In addition, our study suggested that the activity has some relation to students' final grades.

Acknowledgements

This work was supported by JSPS KAKENHI Grant Number JP19H04226.

7. REFERENCES

- [1] R. Aggarwal, N. Gupte, N. Kass, H. Taylor, J. Ali, A. Bhan, A. Aggarwal, S. D. Sisson, S. Kanchanaraksa, J. McKenzie-White, J. McGready, P. Miotti, and R. C. Bollinger. A comparison of online versus on-site training in health research methodology: A randomized study. *BMC Medical Education*, 11(1):1–10, 2011.
- [2] G. Akçapınar, M. N. Hasnine, R. Majumdar, B. Flanagan, and H. Ogata. Developing an early-warning system for spotting at-risk students by

- using eBook interaction logs. *Smart Learning Environments*, 6(1), 2019.
- [3] M. Amin, B. R. Sinha, and P. P. Dey. Online versus onsite teaching performance analysis of an introductory electrical circuit class. *Asian Journal of Education and e-Learning*, 9(4), 2021.
- [4] B. Flanagan and H. Ogata. Learning analytics platform in higher education in Japan. *Journal of Knowledge Management & E-Learning (KM&EL)*, 10(4):469–484, 2018.
- [5] S. Hirokawa and C. Yin. Feature engineering for learning log analysis. *Companion Proceedings 9th International Conference on Learning Analytics & Knowledge (LAK19)*, pages 1–8, 2018.
- [6] S. J. Jones and V. M. Long. Learning equity between online and on-Site mathematics courses. *Journal of Online Learning & Teaching*, 9(1):1–12, 2013.
- [7] LAK22DataChallenge. The 4th Workshop on Predicting Performance Based on the Analysis of Reading Behavior, 2022. <https://sites.google.com/view/lak22datachallenge>.
- [8] S. Leelaluk, T. Minematsu, Y. Taniguchi, F. Okubo, and A. Shimada. Predicting student performance based on lecture materials data using neural network models. In *Proceedings of the 4th Workshop on Predicting Performance Based on the Analysis of Reading Behavior*, 2022.
- [9] H. Ogata, M. Oi, K. Mohri, F. Okubo, A. Shimada, M. Yamada, J. Wang, and S. Hirokawa. Learning analytics for e-book-based educational big data in higher education. In *Smart Sensors at the IoT Frontier*, pages 327–350. Springer, Cham, 2017.
- [10] F. Okubo, A. Shimada, T. Yamashita, and H. Ogata. A neural network approach for students’ performance prediction. In *Proceedings of ACM International Conference on Learning Analytics & Knowledge (LAK)*, pages 598–599, 2017.
- [11] E. Panadero. A review of self-regulated learning: Six models and four directions for research. *Frontiers in Psychology*, 8(APR):1–28, 2017.
- [12] J. Paul and F. Jefferson. A comparative analysis of student performance in an online vs. face-to-Face environmental science course from 2009 to 2016. *Frontiers in Computer Science*, 1(November), 2019.
- [13] A. Shimada, S. Konomi, and H. Ogata. Real-time learning analytics system for improvement of on-site lectures. *Interactive Technology and Smart Education*, 15(4):314–331, 2018.
- [14] A. Shimada, K. Mouri, Y. Taniguchi, H. Ogata, R. I. Taniguchi, and S. Konomi. Optimizing assignment of students to courses based on learning activity analytics. In *Proceedings of the 12th International Conference on Educational Data Mining (EDM)*, pages 178–187, 2019.
- [15] A. Shimada, F. Okubo, C. Yin, and H. Ogata. Automatic summarization of lecture slides for enhanced student preview –Technical report and user study–. *IEEE Transactions on Learning Technologies*, 11(2):165–178, 2018.
- [16] A. Shimada, Y. Taniguchi, F. Okubo, S. Konomi, and H. Ogata. Online change detection for monitoring individual student behavior via clickstream data on e-book system. In *Proceedings of ACM International Conference on Learning Analytics & Knowledge (LAK)*, pages 446–450, 2018.
- [17] X. Yang, D. Li, X. Liu, and J. Tan. Learner behaviors in synchronous online prosthodontic education during the 2020 COVID-19 pandemic. *Journal of Prosthetic Dentistry*, 126(5):653–657, 2021.

Recommendation System of Mobile Language Learning Applications: Similarity versus Diversity in Learner Preference

Juyeong Song
Ewha Womans University
juyeong0910@ewhain.net

Kisu Yang
Korea University
Willow4@korea.ac.kr

Hyo-Jeong So
Dept. of Educational Technology
Ewha Womans University
hyojeongso@ewha.ac.kr

Hyeji Jang
Ewha Womans University
hyejijang@ewhain.net

ABSTRACT

Recognizing the potential of the preference-inconsistent recommendation systems (RS) for learning, this paper aims to examine two recommendation algorithms for mobile language learning applications: RS with similarity and RS with diversity. Diversity was measured through learning styles (how learners learn) and achievement goals (why learners learn). A total of 160 learners participated in the study for building learner profiles and the recommendation algorithm. Overall, our results with RSME indicate that both RS with similarity and RS with diversity performed better than the random recommendation.

Keywords

Recommendation system, Mobile language learning applications, Diversity

1. INTRODUCTION

Recommendation systems (RS) that suggest preferable items intend to help people make proper decisions in a flood of information and choices. In the field of education, researchers have developed RS to recommend learning resources based on ontology [22], learning styles [4], ranked lists of learning objectives [6], and learner preference [5]. Thus far, most educational RS have been based on the algorithm of learners' preferences, recommending items that learners are likely to prefer based on their previous data or learners with similar profiles. Cognitive dissonance theory [10] explains the popularity of this preference-consistent recommendation. When humans process information that does not conform to their existing thoughts and beliefs, they are in a state of cognitive dissonance and experience an uncomfortable psychological state. Hence, for psychological comfort, people tend to only pursue information and items that they prefer to avoid cognitive dissonance.

However, some scholars have started questioning the efficacy of preference-consistent recommendations, arguing that such recommendation methods may reinforce learners' confirmation bias and narrow the opportunity to discover new information and

J. Song, K. Yang, H. Jang, and H.-J. So. Recommendation system of mobile language learning applications: Similarity versus diversity in learner preference. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 758–762, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853141>

experiences [3]. Researchers have proposed 'preference-inconsistent' recommendations that induce discovery learning and higher-order thinking by intentionally providing information that does not match learners' preferences or recommending information that contradicts learners' beliefs [19]. While the preference-inconsistent mechanism has been widely researched in political science and media study to minimize confirmation or information bias, its application in the educational RS is still in its infancy and lacks empirical data to support its efficacy.

Recognizing the potential of the preference-inconsistent recommendation for learning, this paper presents our work-in-progress research that aims to examine two recommendation algorithms for mobile language learning applications (apps): RS with similarity and RS with diversity. Here, 'RS with similarity' refers to the preference-consistent method that recommends the list of apps compatible with learners' learning styles and achievement goals. In contrast, 'RS with diversity' refers to the method that recommends the list of apps that do not match learners' current preferences but have the potential to help learners exposed to new learning styles and achievement goals. It should be noted that RS with diversity does not mean that the system recommends items that learners do not like. Instead, the method is intended to lead learners to discover new items that are likely to have learning efficacy in the future by extending learning experiences beyond a preferred comfort zone.

2. LITERATURE REVIEW

2.1 Recommendation Systems in Education

Recommendation systems have been widely used in the e-commerce area to suggest information and items for potential customers [18]. Recently, RS has been studied in the education sector for various purposes such as suggesting learning activities and courses based on learners' knowledge levels and preferences [16].

From a technical view, there are two main methods of RS: content-based filtering and collaborative filtering. Content-based filtering recommends items that are similar to what users already selected and liked in the past [15]. However, when there is a lack of data on previous behaviors by users, it is hard to recommend appropriate items that match users' preferences. Also, the attributes of items should be well-structured for the accuracy of content-based filtering. In contrast, collaborative filtering recommends items based on similar users' preferences [15]. It is possible to predict items that new users are likely to prefer even if there is a lack of available data

for the target user. Collaborative filtering, however, has a scalability issue. As the number of users or items increases, the complexity of computation becomes higher [11]. Moreover, user-based collaborative filtering systems tend to provide a low number of available items, making final recommendations biased toward popular items.

In the field of technology-enhanced learning (TEL), it has been suggested that there are two perspectives of enhancing filtering techniques: top-down and bottom-up [7]. The top-down perspective focuses on well-defined educational metadata whereas the bottom-up perspective collects user-generated data such as likes, tags, and ratings from learners. In this study, we focus on language learning with mobile applications, which are often used in informal learning contexts. As informal learning is self-directed and less structured with learning goals and time [7], it is difficult to divide applications into specific categories for educational metadata. Hence, this study focuses on the RS from the bottom-up perspective such as learner ratings of respective mobile apps.

2.2 Diversity in Learning Styles and Achievement Goals

Recently, there have been attempts to develop methods for recommending diverse and novel items beyond similarity measures. Described as a ‘long-tail recommendation’, this method recommends diverse and unpopular items in the non-mainstream area to mitigate confirmation bias and to enhance diversity in user experiences [23]. As for movie recommendations, it has been reported that the list of diverse movies positively affects viewers’ satisfaction [14]. However, a long-tail recommendation also suffers from a low accuracy in the recommendation, which necessitates some tradeoff between diversity and accuracy [23].

For learning purposes, there are numerous ways to operationally define diversity. In this study, we focus on diversity in terms of learning styles (*how learners learn*) and achievement goals (*why learners learn*). These two constructs have been well researched and established as a theoretical framework to define learner characteristics in educational research.

First, learning styles refer to the ways learners prefer to receive and process information. According to Felder and Silverman [9], learning styles are divided into four categories: (a) processing (active/reflective), (b) perception (sensing/intuitive), (c) input (visual/verbal), and (d) understanding (sequential/global). Many studies have been conducted on which learning styles are the basis for presenting effective learning methods. However, some scholars suggested two limitations about learning styles [13]. The first limitation is that the preferable way to learn and the actual effective and efficient learning method can be different. The second limitation is that the learning styles instrument showed low reliability [21]. While we are aware of such limitations, this study adopts learning styles as one of the learner-related variables for learner profiles based on the following studies. As for recommender systems, the Felder-Silverman Learning Style Model (FSLSM) has been widely used for two reasons [2, 17]. First, FSLSM can be implemented in a simple way to design personalized e-learning systems [2]. Second, the FSLSM model is appropriate to apply to mobile learning because the learning materials in mobile learning are composed of various formats such as video, text, audio, etc. [17].

Second, achievement goals refer to the aim and focus of one’s actions concerning achievement. Elliot and McGregor proposed the 2X2 achievement goal framework in terms of definition (mastery vs. performance) and valence (approach focus vs. avoidance focus). The framework includes (a) mastery-approach goal, (b)

performance-approach goal, (c) mastery-avoidance goal, and (d) performance-avoidance goal [8]. When learners have the mastery-approach goal, they strive to master learning tasks with the emphasis on learning itself and self-improvement. In contrast, learners with the mastery-avoidance goal tend to avoid misunderstanding or the failure to master a task. Regarding performance, learners with the performance-approach goal concentrate on proving their ability relative to others. They may not be interested in mastering tasks, but doing better than others is more important. On the other, learners with the performance-avoidance goal tend to avoid performing more poorly than others do.

3. RECOMMENDATION SYSTEM

To construct our RS architecture (see Figure 1), three steps were performed: Step 1: constructing the database of RS, Step 2: data analysis, and Step 3: RS evaluation (Experiment). The whole process of the research was conducted with the IRB approval from the researchers’ university.

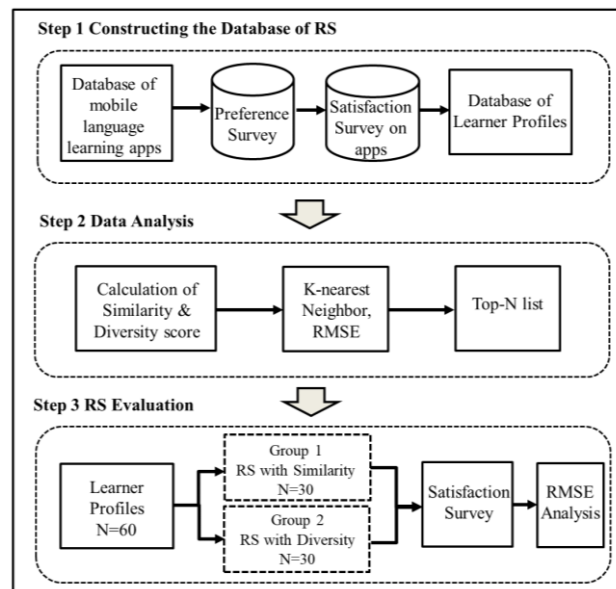


Figure 1. Proposed architecture and process

3.1 Step 1: Constructing the Database of RS

3.1.1 Database of Mobile Language Learning apps

In this study, we focused on recommending mobile language learning apps with artificial intelligence (AI) functions, which have been increasingly developed in the app market. For constructing the database of learner profiles, we chose 18 mobile apps that met five criteria: (1) an application for language learning, (2) an application with AI functions (e.g., voice recognition, adaptive learning system), (3) an application that the developer indicates the use of AI technologies, (4) an application used with no errors, and (5) an application locally available in Korea. Table 1 presents the final list of mobile language learning apps used in this study.

Table 1. List of mobile language learning apps with AI

ID	Application	ID	Application
app 1	Riid Tutor	app 10	Cake
app 2	Super Chinese	app 11	Youbot English Speaking

ID	Application	ID	Application
app 3	Plang	app 12	ELSA Speak: Accent Coach
app 4	Opic up	app 13	Bigple
app 5	Lingo Champ	app 14	Memrise
app 6	Easy Voca	app 15	Mondly
app 7	Say Voca	app 16	AI Tutor
app 8	Youbot Chinese	app 17	Rosetta Stone
app 9	Duolingo	app 18	Busuu

3.1.2 Preference and Satisfaction Survey

To construct the database of learner profiles, we recruited 100 adult learners. Among them, 82 learners fully completed the preference survey that include items about demographic information (e.g., gender, age, education, etc.), learning styles, and achievement goals. For learning styles, we used the ILS (Index of Learning style) instrument with 44 items [9]. For achievement goals, we used the 2X2 Achievement Goal Questionnaire-Revised with 24 items [20]. Then, the participants used four to six mobile language learning apps randomly assigned to them. Finally, they completed the satisfaction survey with 20 items on a 5-point Likert scale to evaluate each app they used. The satisfaction survey includes seven categories: (1) convenience, (2) personalization, (3) professionalism, (4) reliability, (5) appropriateness, (6) satisfaction, and (7) overall satisfaction. We refer to [1] to develop the items related to convenience, personalization, professionalism, and reliability and [12] for the items related to appropriateness of app use and satisfaction. We modified the items according to the context and purpose of this study.

3.1.3 Database of Learner Profiles

To construct the database of learner profiles, data from 82 learners who fully answered the surveys were analyzed based on their learning styles and achievement goals. We generated attribute values as shown in Figure 2. LS1, LS2, LS3, and LS4 refer to four dimensions of learning styles respectively: processing, perception, input, and understanding. LG1, LG2, LG3, and LG4 mean four achievement goals respectively: mastery-approach goal, mastery-avoidance goal, performance-approach goal, and performance-avoidance goal. Since each learner used 4 to 6 apps, the total data of learner profiles is 82*415 (the number of learners * the total number of applications used).

NO.	LS1	LS2	LS3	LS4	LG1	LG2	LG3	LG4
L1	1	0	1	0	4.67	3.00	4.00	1.67
L2	0	1	1	0	3.83	3.67	3.00	3.17
L3	0	0	1	0	5.00	3.00	3.17	1.67
L4	1	1	1	0	3.67	2.50	3.17	2.50
L5	0	1	1	0	4.83	4.17	4.67	1.67

Figure 2. Example of attribute values

3.2 Step 2: Data Analysis

Based on the survey results, we calculated the similarity score for RS with similarity and the diversity score for RS with diversity. The formula for calculating similarity and diversity scores for learner profiles is shown below. We measured the diversity score with Euclidean distance between two learners for learning styles and goals. For example, if the learning styles of learner A and Learner B are [1, 0, 1, 0] and [1, 1, 1, 0] respectively, the diversity score is 1. After calculation, we chose the best similarity measure and the number of K-neighbors.

$$sim(t, t')$$

$$\begin{aligned}
&= \sum_j^2 \lambda_j \cdot sim_j(t, t') \\
&= \lambda_1 \cdot sim_1(t, t') + \lambda_2 \cdot sim_2(t, t') \\
sim_i(t, t') &= diversity\ score * -1 \\
diversity\ score &= Euclidean\ distance(t, t') \\
&= \sqrt{\sum_{i=1}^n (t_i - t'_i)^2}
\end{aligned}$$

To evaluate the performance of RS, this research used Root Mean Squared Error (RMSE), which is widely used to evaluate collaborative filtering systems. From the postulation that users are satisfied with items recommended by diverse users, not just similar ones, RS with diversity returns the top N items sorted by the ascending order of the average scores of K dissimilar users. The diversity score is a negative value of the similarity score, which is the weighted sum of the learning styles similarity and the achievement goals similarity. In the following experiment in Step 3, we set K = 9 and the weights to each similarity $\{\lambda_1, \lambda_2\} = \{1.0, 1.5\}$ because the settings of K and λ show the lowest RMSE between diversity-based prediction and ground truths in training data (see Figure 3).

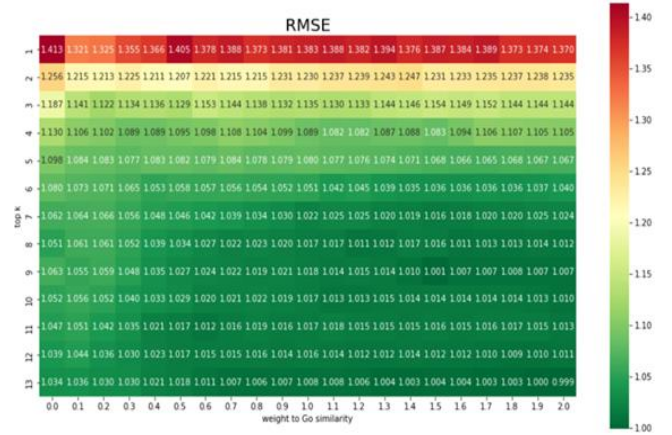


Figure 3. RMSE

3.3 Step 3: RS Evaluation

3.3.1 Experiment

For the experiment to compare the performance of RS with similarity and RS with diversity, we recruited 60 participants aged from 19 to 39 years old. They used the visualized RS (see Figure 4) to complete the same survey administered in Step 1. We analyzed the preference survey data to gauge their learning styles and achievement goals, and then made the recommendations of apps for each participant. We divided 60 participants into two groups with the consideration of their age, job, and gender. The first group was recommended with three apps based on the algorithm of RS with similarity. The second group was recommended with three apps based on the algorithm of RS with diversity. After using the recommended apps, the participants completed the satisfaction survey to evaluate each app they used.

3.3.2 Experimental Setup

For the experiment, we developed the mobile web-based system (see Figure 4) to simulate the real situation of using a RS for learning. The interface shows three buttons to guide learners to access

each page. In the first button ‘What is my preference?’, learners complete the preference survey about demographic information, learning styles, and achievement goals. Clicking the second button ‘Checking recommended apps’, learners see the list of recommended mobile apps for language learning. After using the recommended apps, learners move to the third button ‘What is my satisfaction?’ to complete the survey to indicate their satisfaction with the recommended apps.



Figure 4. RS Visualization

3.3.3 Data Results

We evaluated our recommendation systems with RMSE representing the difference between the predicted and actual levels of learners’ satisfaction. We chose this approach since an accuracy metric only covers the exactly matched values. We observed that RMSE values of similarity-based recommendation and diversity-based recommendation are 0.91 and 1.26 respectively. They are significantly smaller than 1.86 from the random prediction. With this, we interpret that the recommended method that considers how and why learners learn are likely to lead to higher learner satisfaction than the random recommendation. Further, RS with similarity showed a better performance than RS with diversity.

4. LIMITATIONS AND FUTURE WORK

The limitations of this research are as follows. First, the recommendation system was only evaluated by accuracy. Although it is important to predict items accurately, qualitative data such as interviews and proxy indicators (e.g., usage behaviors) should be used additionally to further test and improve the proposed RS. Second, since the data sample is rather small to build a robust model, the research findings have limited generalizability. The small sample size was due to the time and effort required for the participants to use the assigned apps and to provide valid evaluation data. In the future, we plan to recruit more users and expand the database of the apps to build a more robust model. Lastly, while this research compares two RS algorithms, developing a hybrid RS that considers the optimal level of both similarity and diversity is certainly a promising area for future research.

Despite these limitations, we believe that this research makes meaningful contributions to the existing body of literature on RS by providing empirical data that examine similarity versus diversity approaches.

5. ACKNOWLEDGMENTS

This work has been supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2020R1F1A1073469).

6. REFERENCES

- [1] Baek, C. H. 2020. A study on satisfaction of artificial intelligence speaker users using quality evaluation method of artificial intelligence service. *Journal of Korean Service Management Society*. 21, 5 (Dec. 2020), 157-174. DOI=<https://doi.org/10.15706/jksms.2020.21.5.007>.
- [2] Bourkoku, O., El Bachari, E., & El Adnani, M. A. 2016. Personalized e-learning based on recommender system. *International journal of learning and teaching*. 2, 2 (Dec. 2016), 99-103. DOI=<https://doi.org/10.18178/ijlt.2.2.99-103>.
- [3] Buder, J. and Schwind, C. 2012. Learning with personalized recommender systems: A psychological view. *Computers in Human Behavior*. 28, 1 (Jan. 2012), 207-216. DOI=<https://doi.org/10.1016/j.chb.2011.09.002>.
- [4] Chen, H., Yin, C., Li, R., Rong, W., Xiong, Z., and David, B. 2019. Enhanced learning resource recommendation based on online learning style model. *Tsinghua Science and Technology*. 25, 3 (Oct. 2019), 348-356. DOI=<https://doi.org/10.26599/TST.2019.9010014>.
- [5] Dascalu, M.-I., Bodea, C.-N., Mihailescu, M.N., Tanase, E.A., and Ordoñez de Pablos, P. 2016. Educational recommender systems and their application in lifelong learning. *Behaviour & information technology*. 35, 4 (Jan. 2016), 290-297. DOI= 10.1080/0144929X.2015.1128977.
- [6] De Medio, C., Limongelli, C., Sciarrone, F., and Temperini, M. 2020. MoodleREC: A recommendation system for creating courses using the moodle e-learning platform. *Computers in Human Behavior*. 104 (Mar. 2020), 106168. DOI=<https://doi.org/10.1016/j.chb.2019.106168>.
- [7] Drachsler, H., Hummel, H., and Koper, R. 2009. Identifying the goal, user model and conditions of recommender systems for formal and informal learning. *Journal of Digital Information*. 10, 2 (Jan. 2009), 4-24.

- [8] Elliot, A. J. and McGregor, H. A. 2001. A 2 X 2 achievement goal framework. *Journal of Personality and Social Psychology*. 80, 3 (2001), 501-519.
- [9] Felder, F.M. and Soloman, B.A. (1991), Index of Learning Styles. North Carolina State University. Web address: www.ncsu.edu/effective_teaching/ILSdir/ILS-a.htm.
- [10] Festinger, L. 1954. A theory of social comparison processes. *Human relations*. 7, 2 (May. 1954), 117-140. DOI= <https://doi.org/10.1177/001872675400700202>.
- [11] Jalili, M., Ahmadian, S., Izadi, M., Moradi, P., and Salehi, M. 2018. Evaluating collaborative filtering recommender algorithms: a survey. *IEEE access*. 6 (Nov. 2018), 74003-74024. DOI= <https://doi.org/10.1109/ACCESS.2018.2883742>.
- [12] Jang, E., Park, Y., and Lim, K. 2012. Research on factors affecting on learners' satisfaction and purchasing intention of educational applications. *The Journal of the Korea Contents Association*. 12, 8 (Aug. 2012), 471-483. DOI= <https://10.5392/JKCA.2012.12.08.471>.
- [13] Kirschner, P.A. 2017. Stop propagating the learning styles myth. *Computers & Education*. 106 (Mar. 2017), 166-171. DOI= <https://doi.org/10.1016/j.compedu.2016.12.006>.
- [14] Nguyen, T.T., Maxwell Harper, F., Terveen, L., and Konstan, J.A. 2018. User personality and user satisfaction with recommender systems. *Information Systems Frontiers*. 20, 6 (Sep. 2018), 1173-1189. DOI= <https://doi.org/10.1007/s10796-017-9782-y>.
- [15] Ricci, F., Rokach, L., and Shapira, B. 2015. Recommender systems: Introduction and challenges. In *Recommender systems handbook*, F. Ricci, L. Rokach, and B. Shapira, Eds. Springer. Boston, MA, 1-34. DOI= https://doi.org/10.1007/978-1-4899-7637-6_1.
- [16] Santos, O.C. and Boticario, J.G. 2015. User-centred design and educational data mining support during the recommendations elicitation process in social online learning environments. *Expert Systems*. 32, 2 (Jul. 2015), 293-311. DOI= <https://doi.org/10.1111/exsy.12041>.
- [17] Saryar, S., Kolekar, S.V., Pai, R.M., and Manohara Pai, M.M. 2019. Mobile learning recommender system based on learning styles. In *Soft computing and signal processing*, J. Wang, G. Reddy, V. Prasad, and V. Reddy, Eds. Springer. Singapore, SG, 299-312. DOI= https://doi.org/10.1007/978-981-13-3600-3_29.
- [18] Schafer, J.B., Konstan, J., and Riedl, J. 1999. Recommender systems in e-commerce. In *Proceedings of the 1st ACM Conference on Electronic Commerce* (Colorado, USA, November 03-05, 1999). EC'99. ACM, New York, NY, 158-166.
- [19] Schwind, C., Buder, J., Cress, U., and Hesse, F.W. 2012. Preference-inconsistent recommendations: An effective approach for reducing confirmation bias and stimulating divergent thinking? *Computers & Education*. 58, 2 (Feb. 2012), 787-796. DOI= <https://doi.org/10.1016/j.compedu.2011.10.003>.
- [20] Shin, S.-Y. 2007. *Identifying predictability of achievement goal orientation, reward structure and attributional feedbacks on learning achievement in e-PBL*. Doctoral Thesis., Ewha womans university.
- [21] Stahl, S. A. 1999. Different strokes for different folks? A critique of learning styles. *American Educator*. 23, 3 (1999). 98-107.
- [22] Tarus, J.K., Niu, Z., and Mustafa, G. 2018. Knowledge-based recommendation: a review of ontology-based recommender systems for e-learning. *Artificial intelligence review*. 50, 1 (Jan. 2018), 21-48. DOI= <https://doi.org/10.1007/s10462-017-9539-5>.
- [23] Wang, S., Gong, M., Li, H., and Yang, J. 2016. Multi-objective optimization for long tail recommendation. *Knowledge-Based Systems*. 104 (Jul. 2016), 145-155. DOI= <https://doi.org/10.1016/j.knosys.2016.04>.

A Variant of Performance Factors Analysis Model for Categorization

Meng Cao
University of Memphis
mcao@memphis.edu

Philip I. Pavlik Jr.
University of Memphis
ppavlik@memphis.edu

ABSTRACT

Many models of categorization focus on how people form and use knowledge of categories and make predictions about human categorization behaviors [19]. However, few (if any) of them implement these theories into item selection algorithms for category training. The performance Factors Analysis (PFA) model is an alternative to the Bayesian Knowledge Tracing model that tracks students' learning of knowledge components and can be implemented into adaptive practice algorithms [17]. PFA-Difficulty model has been built to select items based on their difficulty level adaptively [4]. This paper describes how we are working to incorporate categorization theories into the PFA model so that it can be used for item selection. We used experiment data of Mandarin tone categorization training to test the model and suggest the implications of the results for item selection.

Keywords

Performance factors analysis, Categorization, Similarity, Item selection.

1. INTRODUCTION

Formal models of categorization make assumptions about human psychological processes during categorization [8,19]. They specify three things in general: (1) the internal representation of the content and format of categorical knowledge, (2) the retrieval process that collects the exact information needed to make a response, (3) the response selection process about how to select a response based on the information collected [2,13]. However, categorization models do not consider individual differences in learning categories and do not track learners' learning process. Therefore, they cannot help instructional design about item selection for training because most processes of learning (other than categorization) are not characterized in the models, e.g., the benefit of active quizzing.

For example, exemplar theory assumes that subjects store each distinct stimulus and its category label in prior memory. To classify a stimulus, subjects compute the similarities between the stimulus representation and all the stored representations of exemplars, aggregate the similarities, and then make a categorical selection [13,19]. But such a strict exemplar theory is not accurate based on what we know about how the brain encodes memories [3]. Analogously, a prototype model operates by computing the similarities between the instances and a summary representation in memory.

M. Cao and P. Pavlik. A variant of performance factors analysis model for categorization. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 763–766, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852974>

The difference is that the prototype model assumes that subjects store a “prototype”, which could be a central tendency of that category instead of every exemplar [13,19]. Importantly, neither of the above models care about training sequence. More recently, Sequential Attention Theory Model (SAT-M) was developed using local context, which considers the influence of the properties of temporally neighboring items during training [5]. However, the model is built based on the Generalized Context Model (GCM) [16], a special case of exemplar models, so it still implausibly supposes a person memorizes all prior instances. Even though it captures the influence of training sequence on the stimuli representation and improves the model fit for data with different training sequences, the model does not intend to give specific suggestions about item selection. The goal of this study is to implement the categorization theories into a learner model so that it can track learners' learning of categories and provide suggestions on item selections for future training.

The Performance Factors Analysis (PFA) model is a student model that predicts individual students on knowledge components (KCs) [1] which are acquired units of cognitive function or structure (e.g., concept, fact, or skill) using counts of successes and failures on prior training trials [17,18]. The PFA model could help us develop adaptive training algorithms. Below are the formulas of the original Performance Factors Analysis model.

$$m(i, j \in KCs, s, f) = \sum_{j \in KCs} (\beta_j + \gamma_j s_{i,j} + \rho_j f_{i,j}) \quad (1)$$

$$p(m) = \frac{1}{1 + e^{-m}} \quad (2)$$

In Equation 1, i represents a student and j represents a KC. m is a logit value representing the accumulated learning of the students on KCs. β_j represents the easiness of the specific KC. s and f count the sum of cases/trials of prior successes and failures. γ and ρ are parameters scaling the effect of the observation counts. In Equation 2, the accumulated learning is converted to probability prediction.

Even though, from the results of the PFA model, we can know whether correct or incorrect responses lead to more learning gains on each KC, the result can't be used for item selection. The reason is that if the success coefficient γ is higher than the failure coefficient ρ , it implies us to select easier items that could achieve immediate success and then maximize the learning gains (the increase of m value) [4]. However, this is in opposition to some pedagogical theories, which suggest that medium-range difficulty of items promotes better learning. For example, the Goldilocks principle in cognitive science [11] and the zone of proximal development [21] suggest that the practices should be neither too simple nor too complex relative to learners' current knowledge. Kelley [12] also stated that keeping an appropriate difficulty level can make training effective. If a task is too easy, it will result in low levels of mental effort, whereas if the task is too difficult, it will be difficult to encode the experience. Both cases lead to low learning gains.

Therefore, we hypothesized that the relationship between difficulty and learning gains could be expressed as an inverted-U function (see equation 3) and incorporate it into the PFA model to build a new model called PFA-Difficulty (see equation 4) [4].

$$y = ax^2 + bx \quad (3)$$

In equation 3, x represents the difficulty of the item. y represents the effect of difficulty on learning gains. In Equation 4, instead of tracking counts of prior successes and failures, we track the effects of prior difficulty levels of successes and failures using quadratic equations.

$$m(i, j \in KCs, s, f) = \sum_{j \in KCs} (\beta_j + \gamma_2 x_{s,i,j}^2 + \gamma_1 x_{s,i,j} + \rho_2 x_{f,i,j}^2 + \rho_1 x_{f,i,j}) \quad (4)$$

We have tried to use the PFA-Difficulty model to select the item with optimal difficulty level for practice. However, the model is not the most appropriate model for categorization. It does not involve learners' categorization process like the prototype and exemplar models do. Furthermore, even though we can treat categories as knowledge components and find the optimal difficulty to learn for each category, there is no suggestion about how to select items between categories. Therefore, this study intended to build a PFA model variant that implements a categorization theory while remaining simple enough to implement for adaptive practice.

2. METHOD

2.1 Dataset

We used a dataset from a Mandarin tone training experiment [4] as a test case for the new model. Two hundred and five participants (Female = 89, Male = 116) who were Amazon Mechanical Turk workers reside in Canada or the US. Age ranged across the lifespan: 10.2% were between 18-25 years old, 34.6% were 26-34, 45.4% were 35-54, and 7.8% were 55-64 years old. Only 2.0% were more than 65 years old. The survey question for education level showed that 12.7% had a high school diploma or GED, 42.0% had some college, 38.5% had a 4-year college degree or bachelor's, and 6.8% had a graduate degree. They finished 216 trials of training in Mandarin tones in the experiment. There are four Mandarin tones (Tone 1, a high-level tone; Tone 2, a rising tone; Tone 3, a low falling-rising tone, and Tone 4, a falling tone). In each trial, participants listened to a tone sound and selected from the four options which tone it was.

2.2 Stimulus representation

The representation of stimuli could be derived from multidimensional scaling (MDS), additive clustering, and factor analysis. Based on the results of previous MDS studies of Mandarin tones [7,10] and the experiment design, we encoded the tone stimuli in 7 features. Three of them are about the F0 (fundamental frequency) direction (level, rising, and down). The other four are about the experiment variables (duration, expansion, and speaker gender).

2.3 PFA-Categorization

Based on exemplar and prototype theories, learners learn through comparing the similarity between the new instance with previous exemplars and prototypes. Here for simplification, we used a prototype mechanism where learners compare the new instance with the prototype of that category. The prototype is calculated by the average feature values of previous examples of the belonging category. According to the SAT-M model, local context is also important. Learners should also learn from comparing the

similarity between the new instance and its previous neighboring item. Therefore, the model should track two types of comparisons: the comparisons with the prototype and the local prior trial.

For the relationship between learning gains and similarity, we have a similar hypothesis to the relationship between learning gains and the difficulty level, which can also be expressed using the simple quadratic function. If the two compared objects are too similar, it will be either too difficult if they belong to different categories or too easy if they are from the same category, and vice versa. Therefore, the medium-range similarity might be optimal for practice. Then the equation of the new PFA-Categorization model can be shown as follows (see equation 5).

$$m(i, j \in KCs, s, f) = \sum_{j \in KCs} (\beta_j + \gamma_2 L_{s,i,j}^2 + \gamma_1 L_{s,i,j} + \rho_2 L_{f,i,j}^2 + \rho_1 L_{f,i,j} + \gamma_4 P_{s,i,j}^2 + \gamma_3 P_{s,i,j} + \rho_4 P_{f,i,j}^2 + \rho_3 P_{f,i,j}) \quad (5)$$

L represents the local comparisons between the current trial and its previous trial by calculating the similarities between the two items for each KC). P indicates the comparisons to the prototype by calculating the similarities between the current trial and its prototype for each KC). γ_1 and γ_2 are scaling the effects of local comparisons of prior successes. ρ_1 and ρ_2 are parameters scaling the effects of local comparisons of prior failures. Similarly, γ_3 and γ_4 and ρ_3 and ρ_4 are parameters scaling the comparisons with prototypes. In this study, for simplification, we used the Euclidean distance as an index of similarity between two items (see equation 6). Parameters p and q represent the to be compared two objects. r is the dimension of the features that range from 1 to n . p_r and q_r are the feature values of objects p and q on dimension r .

$$d(p, q) = [\sum_{r=1}^n (p_r - q_r)^2]^{1/2} \quad (6)$$

2.4 Statistical Analysis

The knowledge components in the Mandarin tone training data are four Mandarin tones. Figure 1 shows an example of the local comparisons and the prototypes' comparisons using four trials of experiment data. The bold solid arrows represent the comparisons, and the dashed arrow indicates the formation of the prototype. The thin solid arrows show the direction of the training sequence. We used the category of previous trials as covariates for the local comparisons when tracking the similarities between the current trial and the previous neighboring trial. The prototype is calculated by averaging the features of previous learned examples of that category. We then used the new PFA-Categorization model to analyze the data and computed the logit function increase or decrease as a function of similarity on each tone. All analyses were completed in R, with source code available from the corresponding author.

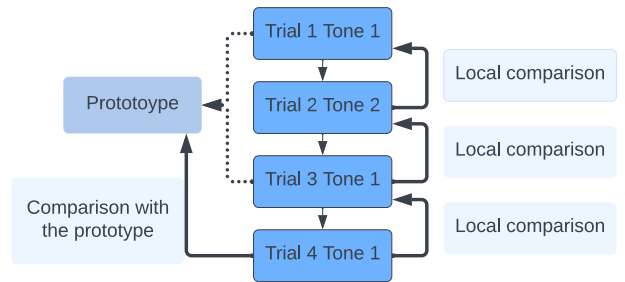


Figure 1. Schematic diagram of the local and prototype comparisons.

3. RESULTS

We did 5-fold cross-validations for the PFA, PFA-Difficulty, and PFA-Categorization models and calculated the mean test fold R^2 (see Table 1). An R package “LKT” (Logistic Knowledge Tracing) [18] was used to do the analysis. In all the models, we used the stimuli (Problem.Name) as the intercept. *Tone* represents the knowledge components (KCs) in the dataset. In the PFA model, *linesuc* and *linefail* track the counts of prior successes and failures of each KC. While in the PFA-Difficulty model, *diffcor1* and *diffcor2* are used to track the effects of difficulty levels of prior successes, and *diffincor1* and *diffincor2* are used to track the effects of difficulty levels of prior failures. Similarly, in the PFA-Categorization model, we used those parameters to track the effects of local similarities and similarities to the prototype. From the R^2 values of the three models, there are minor differences among them, which means they fit the data equally well.

Table 1. Results of logistic regression

Model	Predictors	Mean test fold R^2
PFA	Problem.Name: intercept, Tone: linesuc\$, Tone: linefail\$	0.1663
PFA-Difficulty	Problem.Name: intercept, Tone: diffcor1\$: difficulty, Tone: diffcor2\$: difficulty, Tone: diffincor1\$: difficulty, Tone: diffincor2\$: difficulty	0.1699
PFA-Categorization	Problem.Name: intercept, Tone: diffcor1\$: local, Tone: diffcor2\$: local, Tone: diffincor1\$: local, Tone: diffincor2\$: local, Tone: diffcor1\$: prototype, Tone: diffcor2\$: prototype, Tone: diffincor1\$: prototype, Tone: diffincor2\$: prototype	0.1648

The local comparisons are more closely related to the training sequence than the prototype comparisons. Since the training sequence of the experiment was in random order, there are 16 pairs of local comparisons (4 tones * 4 tones). We used the coefficients of the local comparisons in the PFA-Categorization model to compute the learning efficiency (learning gains divided by the time cost) [9] as a function of the distance levels. The learning gain means the increase or decrease of the m value in equation 5. Table 2 shows an example of the learning gains calculation when Tone 1 is the previous trial and Tone 1, Tone 2, Tone 3, or Tone 4 is the current trial. The distance value was normalized to be in a range of 0 and 1, which was initially between 0 and 5.76.

Table 2. The function logit change (learning gains) of distance

Local comparisons	The function logit change
Tone 1-Tone 1	$x*(0.30*x-0.30*(x)^2) + (1-x)*(-0.33*x+0.30*(x)^2)$
Tone 1-Tone 2	$x*(-0.12*x+0.20*(x)^2) + (1-x)*(-0.03*x+0.12*(x)^2)$
Tone 1-Tone 3	$x*(0.19*x-0.08*(x)^2) + (1-x)*(-0.17*x+0.18*(x)^2)$
Tone 1-Tone 4	$x*(0.21*x+0.00*(x)^2) + (1-x)*(-0.29*x+0.22*(x)^2)$

Then we plotted the learning efficiency of the local comparison pairs and found the maximum values of logit change per second given the distance between two tones. For example, Figure 2 shows the learning efficiency curves of local comparisons when Tone 1 is the previous tone, and the adjacent current tone could be Tone 1, Tone 2, Tone 3, or Tone 4. For the Tone 1-Tone 1 pair, the logit gain from trial per second is the largest when the distance between them is 0.83, and the maximal learning gain per second is 0.003. For Tone 1-Tone 2 pair, Tone 1-Tone 3, and Tone 1-Tone 4, the optimal distance values are 1, 1, and 1, and the maximal learning gain is 0.014, 0.019, 0.037, respectively. Therefore, to achieve the maximal learning gain when practicing Tone 1, we should select Tone 4 as the next learned item. Similarly, after analyzing Tone 2, Tone 3, and Tone 4 as the previous trial separately, the findings suggest that we select Tone 4, Tone 1, and Tone 3, respectively, to achieve maximal learning gain. Above findings are just the first step to build an adaptive training system since we only have a static prediction about which tone category to select. The predictions are not sensitive to learners’ performance changes due to learning. Adaptive training not only needs to consider individual differences (learning rate) but also needs to adjust the similarity between practice trials accordingly to achieve the maximal learning gain for individuals. That is the direction for future work which is out of the scope of this preliminary report.

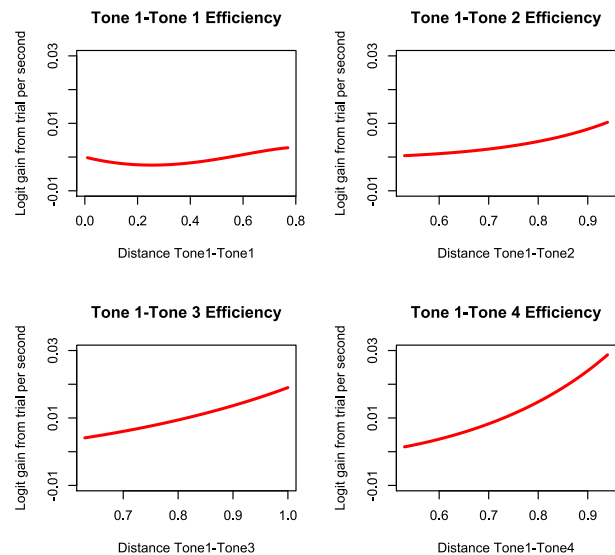


Figure 2. Plots of logit change given the distance between two tones when Tone 1 is the previous trial.

4. DISCUSSION

In this study, we are combining categorization theories with the PFA model to track learners’ learning of categories. Instead of tracking counts of prior successes and failures as the PFA model does or effects of difficulty levels of prior successes and failures as the PFA-Difficulty model does, the new PFA-Categorization model tracks the similarities between the adjacent trials and the similarity between each trial and its prototype of the belonging category. The R^2 values showed that the performance of the PFA-Categorization model was better than the original PFA model. Even though the quantitative comparison did not reveal differences, it has useful implications for later item selection for category training. After learning a specific tone category, it may be possible to make decisions about what category and item are to be practiced next.

Many studies have examined the influence of presentation order of examples on category learning [6,14,20]. For example, Carvalho and Goldstone [6] suggest that if stimuli have high within- and between-category similarity interleaved study could result in better generalization, whereas if categories have low within- and between-category similarity, blocked presentation can lead to better generalization. However, their study used extreme examples of similarities that are either too similar or too dissimilar, which is not so often seen in regular category learning. They also have no systematic suggestion about how we should sequence the categories based on their similarity levels and learning gains. Prior studies have simple manipulations about similarity either maximize or minimize the similarity between adjacent training trials [14,15]. These methods are too general and may not fit all the category learning cases. The benefit of the PFA-Categorization model is that it could give us suggestions about what we should select next based on the analysis of the training data to achieve maximal learning efficiency.

5. CONCLUSION

This paper implemented a categorization mechanism into the PFA model to track category learning that suggests what category we should select next based on the learning efficiency curve as a function of the similarity between the two adjacent items. We plan to use a developed version of the model in an adaptive training system to select the item with optimal similarity levels for learners to achieve maximal learning efficiency.

Future work will improve the model by adding weights of features into the distance calculation since different features may have different importance for categorization. We will also test the model with more datasets since the data used in the present study did not have any manipulation of practice schedules so it may not work well for the data. For example, interleaved practice and blocked practice may have different types of local comparisons that the former is mainly between-category comparisons, and the latter is within-category comparisons. However, since the practice data we used mixed between and within category comparisons randomly it is difficult to see the cumulative effects of scheduling strategy in the data.

6. REFERENCES

- [1] Aleven, V., & Koedinger, K. R. 2013. Knowledge component (KC) approaches to learner modeling. *Design Recommendations for Intelligent Tutoring Systems, 1*, 165-182.
- [2] Ashby, F. G., & Maddox, W. T. 1993. Relations between prototype, exemplar, and decision bound models of categorization. *Journal of Mathematical Psychology*, 37(3), 372-400.
- [3] Bowman, C. R., Iwashita, T., & Zeithamova, D. (2020). Tracking prototype and exemplar representations in the brain across learning. *Elife*, 9, e59360.
- [4] Cao, M., Pavlik Jr, P. I., & Bidelman, G. M. 2019. Incorporating Prior Practice Difficulty into Performance Factor Analysis to Model Mandarin Tone Learning. In Lynch, C. F., Merceron, A., Desmarais, M., & Nkambou, R. (Eds.), *Proceedings of the 12th International Conference on Educational Data Mining* (pp. 516 - 519). Montreal, Canada.
- [5] Carvalho, P. F. & Goldstone, R. 2019. A computational model of context-dependent encodings during category learning. *PsyArxiv*. <https://doi.org/10.31234/osf.io/8psa4>.
- [6] Carvalho, P.F. and Goldstone, R.L. 2014. Putting category learning in order: Category structure and temporal arrangement affect the benefit of interleaved over blocked study. *Memory & Cognition*. 42, 3 (Apr. 2014), 481–495. DOI:<https://doi.org/10.3758/s13421-013-0371-0>.
- [7] Chandrasekaran, B., Sampath, P. D., & Wong, P. C. 2010. Individual variability in cue-weighting and lexical tone learning. *The Journal of the Acoustical Society of America*, 128(1), 456-465.
- [8] Chandrasekaran, B., Koslov, S.R. and Maddox, W.T. 2014. Toward a dual-learning systems model of speech category learning. *Frontiers in Psychology*. 5, 825. DOI:<https://doi.org/10.3389/fpsyg.2014.00825>.
- [9] Eglington, L. G., & Pavlik Jr, P. I. 2020. Optimizing practice scheduling requires quantitative tracking of individual item performance. *NPJ science of learning*, 5(1), 1-10.
- [10] Gandour, J. T. 1978. Perceived dimensions of 13 tones: a multidimensional scaling investigation. *Phonetica*, 35(3), 169-179.
- [11] Halpern, D.F., Graesser, A. and Hakel, M., 2008. Learning principles to guide pedagogy and the design of learning environments. *Washington, DC: Association of Psychological Science Taskforce on Lifelong Learning at Work and at Home*.
- [12] Kelley, C.R., 1969. What is adaptive training? *Human Factors*, 11(6), pp.547-556.
- [13] Kruschke, J. K. 2008. Models of categorization. In R. Sun (Ed.), *The Cambridge handbook of computational psychology* (Cambridge Handbooks in Psychology, pp. 267-301). Cambridge:Cambridge University Press. <https://doi.org/10.1017/CBO9780511816772.013>.
- [14] Mathy, F., & Feldman, J. 2009. A rule-based presentation order facilitates category learning. *Psychonomic bulletin & review*, 16(6), 1050-1057.
- [15] Medin, D. L., & Bettger, J. G. 1994. Presentation order and recognition of categorically related examples. *Psychonomic bulletin & review*, 1(2), 250-254.
- [16] Nosofsky, R. M. 1986. Attention, similarity, and the identification-categorization relationship. *Journal of experimental psychology: General*, 115(1), 39.
- [17] Pavlik Jr, P. I., Cen, H., & Koedinger, K. R. 2009. Performance factors analysis -- A new alternative to knowledge tracing. In V. Dimitrova, R. Mizoguchi, B. d. Boulay, & A. Graesser (Eds.), *Proceedings of the 14th International Conference on Artificial Intelligence in Education* (pp. 531–538). Brighton, England.
- [18] Pavlik, P. I., Eglington, L. G., & Harrell-Williams, L. M. (2021). Logistic Knowledge Tracing: A Constrained Framework for Learner Modeling. *IEEE Transactions on Learning Technologies*, 14(5), 624-639.
- [19] Pothos, E. M., & Wills, A. J. (Eds.). 2011. *Formal approaches in categorization*. Cambridge University Press.
- [20] Sandhofer, C. M., & Doumas, L. A. 2008. Order of presentation effects in learning color categories. *Journal of Cognition and Development*, 9(2), 194-221.
- [21] Vygotsky, L., 1986. *Thought and Language*. MIT Press, Cambridge, Mass.

Selecting Reading Texts Suitable for Incidental Vocabulary Learning by Considering the Estimated Distribution of Acquired Vocabulary

Yo Ehara
Tokyo Gakugei University
ehara@u-gakugei.ac.jp

ABSTRACT

In second language acquisition, incidental vocabulary learning refers to the process by which one's vocabulary increases through activities in which increasing vocabulary is not the main goal. A typical example is extensive reading, where learners naturally expand their vocabulary by reading many texts and guessing the meanings of unfamiliar words. Selecting texts suitable for incidental learning requires a personalized and fine-grained estimation of each language learner's vocabulary. If a learner does not know sufficient words in a text, then the learner cannot read through the text to guess the meanings of unfamiliar terms, so incidental learning does not occur. In contrast, if a learner knows all the words in a text, then incidental learning cannot occur because the learner has no new words to learn. Therefore, if a learner attempts to select a text that can significantly increase their vocabulary, the risk of reading failure increases along with the possibility that no words can be guessed and learned. Therefore, learners should be presented with both the amount of vocabulary they can add and risk of failure in reading, and be allowed to select the text they wish to read. To this end, we require an algorithm that can simultaneously calculate the amount of vocabulary that can be learned and relevant risk when a text is read. This paper presents an algorithm for this purpose with preliminary experimental results. Specifically, we use findings from applied linguistics that indicate that the condition for incidental learning to occur is that the percentage of words that a learner knows in a text is above a certain threshold. By modeling the estimated size of the increase in vocabulary as a random variable, our method uses the variance of the estimated size as a measure of the risk of reading failure. This allows a learner to select the text with the lowest risk among texts that have the same estimated size of increase in vocabulary. Experimental results demonstrate that our method can significantly aid learners in selecting "efficient" texts to read by identifying a handful of such texts among a library of 500 texts. The results also demonstrate that some texts are stable and efficient for many learners.

Y. Ehara. Selecting reading texts suitable for incidental vocabulary learning by considering the estimated distribution of acquired vocabulary. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 767–772, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852976>

Keywords

Second Language Vocabulary, Extensive Reading, Natural Language Processing

1. INTRODUCTION

In second language acquisition, the "incidental learning" of second language vocabulary is the incidental acquisition of vocabulary through activities in which vocabulary acquisition is not the main objective. It is a concept that is widely used in applied linguistics studies [20, 28, 11]. An example of such an activity is *extensive reading*, where readers learn vocabulary by guessing the meanings of unfamiliar words when reading various texts. In contrast to incidental learning, learning vocabulary in the context of an activity whose main goal is vocabulary acquisition is called "intentional learning" [28, 11]. An example is acquiring second language vocabulary by memorizing lists of words. Although incidental learning is less efficient than intentional learning at increasing learner vocabulary in a second language, it is believed to have many merits. It promotes a deeper understanding of how words are used in different contexts and leads to improved learner writing skills [21].

Being able to select texts of interest to each learner also improves motivation for language learning. These merits are summarized in [21] and a combination of intentional and incidental learning is desirable [21].

Incidental learning has many advantages and is used in various scenarios. In most scenarios, for successful incidental learning, it is important to select and use texts that suit the learner [21] and various technologies that support incidental learning have been studied [9, 10, 12]. New vocabulary acquisition does not occur if a text consists only of words that have already been learned. In contrast, if a learner cannot understand sufficient words in a text, then there is a high risk of *failure* in reading the text, making incidental learning difficult because the learner cannot guess the meanings of words appearing in the text. In this case, even if we allow learners to use dictionaries, they are not motivated to continue learning if there are too many words that they need to look up [27, 18]. In applied linguistics, it has been reported that if a learner knows 95% to 98% of the words (tokens) in a text, then they can learn new vocabulary through incidental learning by guessing the meanings of unfamiliar words based on their context [13]. In other words, this finding from applied linguistics suggests that unlearned words in a text can be efficiently learned incidentally if a learner knows 95% to

98% of the text’s tokens [21].

This result from applied linguistics implies that text selection for *efficient incidental learning should consider the risk of failure when reading the selected text*. Selecting a text with a small ratio of known words (i.e., a large ratio of unknown words) may increase the size of the vocabulary acquired through incidental learning, but if the ratio of known words is below a certain threshold, then learners will not be able to continue reading the text and guess the meanings of unfamiliar words. Therefore, incidental learning will not occur and the size of vocabulary acquired will not increase. Therefore, it is desirable to select texts that appear to have the largest number of unknown words while keeping the risk of learner reading failure within an acceptable range.

In this study, we mathematically formulated a strategy for increasing vocabulary through incidental learning using machine learning. Specifically, we propose a method for determining the probability distribution of the estimated size of the vocabulary acquired through incidental learning when a learner reads a text. We modelled the variance of the estimated size as a measure of the risk of reading failure. By using our method, among many texts, learners can select a text for incidental learning by considering both the number of words to be learned (i.e., estimated size of the increase in vocabulary) and risk of failure in reading the text.

2. RELATED WORK

In educational data mining and intelligent tutoring system studies, technologies for supporting incidental learning have been studied in lifelogs [9] and dictionaries [10, 12]. However, previous studies have not focused on methods that can consider both the estimated size of the increase in vocabulary and risk of reading failure. Several researchers have also proposed methods to assess second language learner vocabularies [19, 23] and tutoring systems [26]. However, to the best of our knowledge, no study has extensively addressed text selection algorithms based on the estimated increase in second language vocabulary through incidental learning.

This study focused on the task of *personalized prediction* of second language vocabulary [16, 4, 15, 29, 8, 14, 6, 7], which has been studied in natural language processing. In this task, each learner takes a vocabulary test that can be completed in approximately half an hour. Then, for a word *not on the test*, this task aims to predict whether each test taker knows the word based on their vocabulary test results. Some previous studies have used rich information other than vocabulary tests to improve accuracy, such as individual learner characteristics and word meanings [7, 4], as well as human memory lengths [25, 24].

In particular, several studies have modeled the degree of forgetting over time considering that forgetting curves are closely related to vocabulary learning [25, 24]. However, such meticulous modeling requires time-stamped records of vocabulary learning processes, which are only provided in a few datasets. We used a vocabulary test result dataset to test the applicability of our method.

3. TASK SETTING

Here, we present a motivating example for the problem considered in this study. Consider a text consisting of 4 words [a,b,c,d]. Let the frequency of the words be [93, 3, 3, 1]. Assume that the probability of knowing each word [a,b,c,d] is [0.9,0.6,0.5,0.2], respectively. In reality, these probabilities are estimated using probabilistic classifiers trained on a vocabulary test that includes *none* of a, b, c, or d. Therefore, we assume that we cannot directly test whether a learner knows these words.

According to findings from applied linguistics, if a learner knows more than 95% of the words in a text, they can learn unfamiliar words by reading the text. This percentage is called *text coverage*. Here, we enumerate the cases in an example where text coverage exceeds 95%. In this example, “a” occurs the most frequently, but knowing “a” alone does not ensure that the text coverage exceeds 95%. However, if the learner knows {a,b} and does not know {c,d}, then the coverage of the learner for the target text exceeds the threshold. If we focus only on the words that the learner knows and writes specifically, then the cases that exceed the threshold are {a,b}, {a,c}, {a,b,d}, {a,c,d}, and {a,b,c,d}.

Now, consider a case in which the learner knows words {a,b}. Because we assume that the learner does not know {c,d}, the probability of case {a,b} can be calculated as $0.9 \times 0.6 \times (1 - 0.5) \times (1 - 0.2)$. At this time, the aforementioned applied linguistics finding [20, 22] indicates that this learner can naturally acquire the unfamiliar words {c,d} by reading the text. If we focus only on unfamiliar words, we can consider this probability as *the probability that the learner can learn the words {c,d} by reading the text* ($0.9 \times 0.6 \times (1 - 0.5) \times (1 - 0.2)$) through incidental learning. Similarly, the probability that the learner knows only {a,c} can also be considered as the probability that new words {b,d} can be acquired by reading the text through incidental learning.

Therefore, when considering all cases, the probability that the learner can incidentally learn each word [a,b,c,d] by reading the text is [0, 0.18, 0.27, 0.576]. If the learner does not know “a,” then they cannot read the text and incidental learning cannot occur. Therefore the probability for “a” is 0. The expected number of words that can be acquired by reading this text is $1 \times 0 + 1 \times 0.18 + 1 \times 0.27 + 1 \times 0.576 = 1.026$. In the next section, we propose a method for not only obtaining the expectation of learning (i.e., mean of the distribution), but also the entire distribution.

4. PROPOSED FORMULATION

We formulated a motivating example in the previous section. Consider a vocabulary $\{v_1, \dots, v_I\}$, where I is the vocabulary size. Let n_i be the number of occurrences of v_i (i.e., frequency of v_i in the text).

Assume that we focus on a specific learner among all J learners. We denote the probability that the learner knows v_i as p_i . Let the threshold value of text coverage be τ . In the motivating example presented in the previous section, $\tau = 0.95$. The probability of the text coverage exceeding this threshold can be expressed as follows. First, the total number of words in a text can be expressed as $N = \sum_{i=1}^I n_i$. Next, consider the following random variable Z_i , which is equal to 1 if the learner knows the word v_i and equal to zero 0

otherwise. $\{Z_1, \dots, Z_I\}$ are assumed to be independent of each other.

$$Z_i \sim \text{Bernoulli}(p_i) \quad (1)$$

Because the number of occurrences of words known to the learner in the text is $\sum_{i=1}^I Z_i n_i$, then text coverage can be expressed as $\frac{\sum_{i=1}^I Z_i n_i}{N}$. Therefore, the probability that the text coverage exceeds the threshold is $P(\sum_{i=1}^I Z_i n_i \geq N\tau)$.

The probability that the learner acquires a new word v_i through incidental learning by reading the text can be formulated as follows. For incidental learning to occur, the text must be readable, so text coverage must be above the threshold. Furthermore, for a word v_i to be learned, the learner must *not* already know the word v_i . Therefore, this probability can be expressed as $P(Z_i = 0, \sum_{i=1}^I Z_i n_i \geq N\tau)$, which is hereafter denoted as q_i .

We wished to obtain the distribution of the number of words acquired through incidental learning from a text. Therefore, we define A_i as

$$A_i \sim \text{Bernoulli}(q_i). \quad (2)$$

Here, $\{A_1, \dots, A_I\}$ are assumed to be independent of each other. The number of acquired words, which is denoted as A , can be expressed as $A = \sum_{i=1}^I A_i$. Because each A_i is a random variable, A is also distributional.

5. SOLVING THE PROBLEM

To obtain the probability desired $P(\sum_{i=1}^I Z_i n_i \geq N\tau)$, or the probability that the text coverage exceeds the threshold, and to obtain the distribution of the number of acquired words $A = \sum_{i=1}^I A_i$, we must find the probability of a random variable consisting of a sum of independent binomial distributions with *different success probabilities*. If the success probabilities of the distributions are equal, then their sum is also a binomial distribution based on the reproducing property of the binomial distribution. However, in this case, because the success probabilities are different, the sum does not follow a binomial distribution. This distribution is called a *Poisson binomial distribution*.

Regarding the second language vocabulary, in order to calculate $P(\sum_{i=1}^I Z_i n_i \geq N\tau)$, [5] proposed a method for obtaining this type of probability using dynamic programming. Based on space limitations, we do not present their algorithm in detail in this paper. Instead, we present only an outline of their algorithm. Because $\sum_{i=1}^I Z_i n_i$ is an integer, the condition that $N\tau$ or more can be attributed to the subset-sum problem. Given a target integer, the goal of the subset-sum problem is to determine if there is a subset of $\{n_1, \dots, n_I\}$ that adds up to the target integer. The subset-sum problem can be solved using dynamic programming (DP). Specifically, we can use a DP table consisting of integers that can be created using subsets of $\{n_1, \dots, n_I\}$. Additionally, we must determine $P(Z_i = 0, \sum_{i=1}^I Z_i n_i \geq N\tau)$. This is computed by applying an extension to each cell of the DP table to record the probability value of $\{Z_1, \dots, Z_I\}$. In the Poisson binomial distribution, $\sum_{i=1}^I p_i$ is the mean and $\sum_{i=1}^I p_i(1 - p_i)$ is the variance. We used this fact to obtain the mean and variance values in our experiments.

6. EXPERIMENTS

For learner vocabulary test results, [4] published a dataset based on crowdsourcing. We adopted this dataset for our experiments. According to [4], in this dataset, the vocabulary size test (VST) [1], which consists of 100 vocabulary questions, was completed by 100 learners who had taken the TOEIC test (<https://www.ets.org/toEIC>) using a Japan-based crowdsourcing service called Lancers. The VST is a multiple-choice test in which the test taker selects the appropriate paraphrase of a word *within an English sentence* from four options. To avoid questions being solved based on grammatical clues that are irrelevant to knowledge of the word such as the difference between singular and plural words, all options are designed to be grammatically correct if they are replaced with the word in question.

Using this dataset, we trained a probabilistic classifier to classify whether each learner knows a given word and used the resulting probability values as p_i in Eq. (1). Because this is a binary classification problem, a neural classifier can be used. However, because improving classification performance was not the goal of this study, simple logistic regression was used to train the classifier. For the features of the classifier, we used the frequencies from the Corpus of Contemporary American English [3] and the British National Corpus [2]. The frequencies were converted into $-\log(\text{frequency})$ values and used as features. To make personalized predictions in which the prediction results differ from learner to learner, we simply added a J -dimensional one-hot vector as a feature, where J is the number of learners considered in the model. For this purpose, we followed the method presented by [5].

The Brown corpus was used as the set of texts for selection. To ensure that the lengths of the texts did not affect the experimental results, for each of the 500 texts in the Brown corpus, the first 300 tokens of the text were used in our experiments. From these 500 texts, our goal was to select one suitable for a learner's incidental learning.

In this setting, incidental learning is likely to occur in high-performing learners because low-performing learners can read few texts from the Brown corpus. Therefore, we conducted our experiments based on the highest-performing learner, who correctly answered 96 of the 100 questions in the VST. We randomly selected a text from the Brown corpus and estimated the distribution of the vocabulary size acquired by this learner through incidental learning when reading the text shown in Fig. 1. This distribution is more informative compared to when only the mean is available.

The value and variance of the expected acquired vocabulary when reading each text are considered simultaneously in Fig. 2. Each point represents a text in the Brown corpus. For a given expected size of the acquired vocabulary, it is better for the learner to select the text with the lowest possible variance in the acquired vocabulary. This allows learners to increase their vocabulary with a lower risk of reading failure. The upper-left portion of Fig. 2 presents the most *efficient* set of texts that promotes incidental learning for this learner. Therefore, the vertical axis in Fig. 2 can be considered as the gain and the variance of the acquired vocabulary in the horizontal axis can be considered as the *risk*. Fig. 2 can be considered as a risk–return plot, which is often

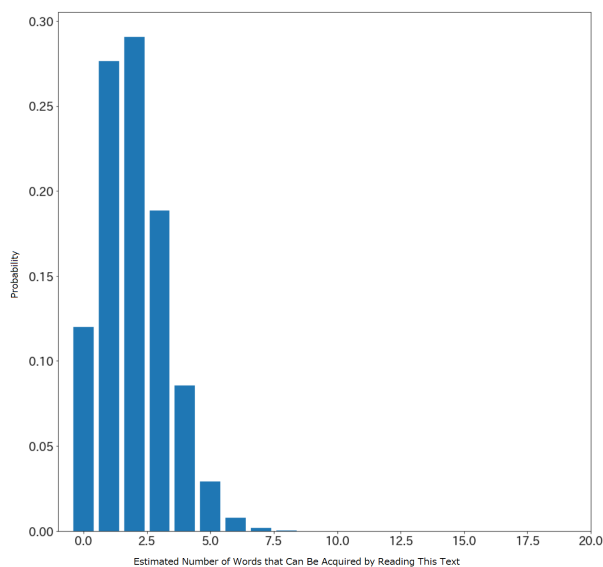


Figure 1: Example of the distribution of the estimated size of the acquired vocabulary. The horizontal axis represents the number of words that the learner is expected to acquire by reading the text and the vertical axis represents the probability.

used in finance [17].

In a risk–return plot such as in **Fig. 2**, selecting the upper-left points yields the lowest risk and highest gain. Therefore, the green dashed line created by connecting the upper-left points is called the *efficient frontier*. In **Fig. 2**, among the 500 texts, 5 texts on the efficient frontier were selected. In other words, we were able to narrow down the number of texts suitable for the learner’s incidental learning by a factor of 1/100. The choice of which of these 5 texts to select depends on the degree of risk that is acceptable for learners to increase their vocabulary. Although selecting one text among 500 is difficult for a human, selecting one text among 5 is relatively easy.

An interesting question is whether there texts that are efficient for most learners. An experiment was conducted to answer this question. **Fig. 2** presents the efficient frontier for the learner with the highest score in the vocabulary test dataset. Similarly, we identified the top 30 highest-performing learners in a dataset consisting of 100 learners.

Tab. A.1 presents the results. One can see that 7 texts were selected from the efficient frontiers of more than 10 learners, with the most common text being included in the efficient frontiers of 14 learners. These results indicate that the texts included in the efficient frontiers of learners are relatively stable, even though efficient frontiers vary by learner.

Although we leave the quantitative analysis of the texts in **Tab. A.1** for future work, the following qualitative trends can be observed. The top text (first line in **Tab. A.1**) exhibits clear “contrasts.” In other words, difficult words such as “contrarities,” which many English learners presumably do not know, occur with words that they are likely to know

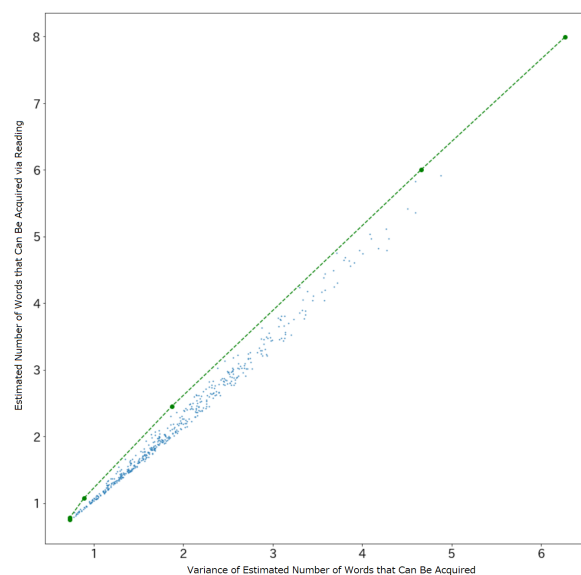


Figure 2: The mean (vertical) and variance (horizontal) of the estimated size of the vocabulary acquired for each text (i.e., each dot). Because our method is personalized, this figure is tailored to each learner. In this figure, we present the case of the top learner in the vocabulary test dataset as an example.

such as “belief” and “imagination.” However, because proper nouns are treated as out-of-vocabulary words, regardless of the learner, texts with many proper nouns are also likely to be listed as efficient texts by most learners in **Tab. A.1**, as shown in the second row of **Tab. A.1**. Addressing this factor is a topic for future study.

7. CONCLUSION

To select texts suitable for vocabulary acquisition through incidental learning, we proposed a method for calculating estimates of the vocabulary acquired through incidental learning for each pair of learners and texts based on the findings of applied linguistics. By considering the size of the vocabulary acquired as a type of *gain* for a learner, we introduced the concept of the *efficient frontiers* used in financial engineering for vocabulary learning. We experimentally showed that there are “efficient” texts suitable for incidental learning that are included in the efficient frontiers of many learners.

The introduction of the concept of efficient frontiers to language learning will enable vast future work in this field, including evaluation experiments over time using learning application logs, handling the aforementioned proper noun issue, and multi-text selection when reading multiple texts simultaneously. Our approach can also be used with modern portfolio theory [17] to identify the most efficient texts.

8. ACKNOWLEDGMENTS

This work was supported by JST ACT-X Grant Number JPMJAX2006 in Japan. We used the AIST ABCI infrastructure and RIKEN miniRaiden system for computational resources. We appreciate the valuable comments from anonymous reviewers.

9. REFERENCES

- [1] D. Beglar and P. Nation. A vocabulary size test. *The Language Teacher*, 31(7):9–13, 2007.
- [2] T. BNC Consortium. *The British National Corpus, version 3 (BNC XML Edition)*. 2007.
- [3] M. Davies. The 385+ million word corpus of contemporary american english (1990–2008+): Design, architecture, and linguistic insights. *Intl. j. of corp. ling.*, 14(2):159–190, 2009.
- [4] Y. Ehara. Building an English Vocabulary Knowledge Dataset of Japanese English-as-a-Second-Language Learners Using Crowdsourcing. In *Proc. of LREC*, May 2018.
- [5] Y. Ehara. Uncertainty-Aware Personalized Readability Assessments for Second Language Learners. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pages 1909–1916, Dec. 2019.
- [6] Y. Ehara, Y. Baba, M. Utiyama, and E. Sumita. Assessing Translation Ability through Vocabulary Ability Assessment. In *Proc. of IJCAI*, 2016.
- [7] Y. Ehara, Y. Miyao, H. Oiwa, I. Sato, and H. Nakagawa. Formalizing Word Sampling for Vocabulary Prediction as Graph-based Active Learning. In *Proc. of EMNLP*, pages 1374–1384, 2014.
- [8] Y. Ehara, I. Sato, H. Oiwa, and H. Nakagawa. Mining Words in the Minds of Second Language Learners for Learner-specific Word Difficulty. *Journal of Information Processing*, 26:267–275, 2018.
- [9] M. N. Hasnine, K. Mouri, G. Akcapinar, M. M. H. Ahmed, and H. Ueda. A new technology design for personalized incidental vocabulary learning using lifelog image analysis. In *Proceedings of the 28th international conference on computers in education (ICCE2020)*, pages 516–521, 2020.
- [10] M. Hill and B. Laufer. Type of task, time-on-task and electronic dictionaries in incidental vocabulary acquisition. 2003.
- [11] J. H. Hulstijn. Intentional and incidental second-language vocabulary learning: A reappraisal of elaboration, rehearsal and automaticity. 2001.
- [12] B. Laufer and J. Hulstijn. Incidental vocabulary acquisition in a second language: the construct of task-induced involvement. *Applied Linguistics*, 22(1):1–26, Mar. 2001.
- [13] B. Laufer and G. C. Ravenhorst-Kalovski. Lexical Threshold Revisited: Lexical Text Coverage, Learners’ Vocabulary Size and Reading Comprehension. *Reading in a Foreign Language*, 22(1):15–30, Apr. 2010.
- [14] J. Lee and C. Y. Yeung. Automatic prediction of vocabulary knowledge for learners of Chinese as a foreign language. In *2018 2nd International Conference on Natural Language and Speech Processing (ICNLSP)*, pages 1–4, Apr. 2018.
- [15] J. Lee and C. Y. Yeung. Personalizing Lexical Simplification. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 224–232, Santa Fe, New Mexico, USA, Aug. 2018. Association for Computational Linguistics.
- [16] J. Lee and C. Y. Yeung. Personalized Substitution Ranking for Lexical Simplification. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 258–267, Tokyo, Japan, Oct. 2019. Association for Computational Linguistics.
- [17] H. M. Markowitz. Foundations of portfolio theory. *The journal of finance*, 46(2):469–477, 1991. Publisher: JSTOR.
- [18] A. Mikami. Students’ attitudes toward extensive reading in the japanese efl context. *Tesol Journal*, 8(2):471–488, 2017.
- [19] B. Naismith, N.-R. Han, A. Juffs, B. Hill, and D. Zheng. Accurate Measurement of Lexical Sophistication in ESL with Reference to Learner Data. In *Proceedings of the 11th International Conference on Educational Data Mining*, pages 259–265, Buffalo, NY, USA, 2018. International Educational Data Mining Society (IEDMS).
- [20] T. Nakata and I. Elgort. Effects of spacing on contextual vocabulary learning: Spacing facilitates the acquisition of explicit, but not tacit, vocabulary knowledge. *Second Language Research*, 37(2):233–260, 2021. eprint: <https://doi.org/10.1177/0267658320927764>.
- [21] I. Nation and R. Waring. *Teaching extensive reading in another language*. Routledge, 2019.
- [22] P. Nation. How much input do you need to learn the most frequent 9,000 words? 2014. Publisher: University of Hawaii National Foreign Language Resource Center.
- [23] R. Paiva, I. I. Bittencourt, W. Lemos, A. Vinicius, and D. Dermeval. Visualizing Learning Analytics and Educational Data Mining Outputs. In C. Penstein Rosé, R. Martínez-Maldonado, H. U. Hoppe, R. Luckin, M. Mavrikis, K. Porayska-Pomsta, B. McLaren, and B. du Boulay, editors, *Artificial Intelligence in Education*, pages 251–256, Cham, 2018. Springer International Publishing.
- [24] S. Reddy, S. Levine, and A. Dragan. Accelerating human learning with deep reinforcement learning. In *NIPS workshop: teaching machines, robots, and humans*, 2017.
- [25] B. Settles and B. Meeder. A Trainable Spaced Repetition Model for Language Learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1848–1858, Berlin, Germany, Aug. 2016. Association for Computational Linguistics.
- [26] A. Sinclair, K. McCurdy, C. G. Lucas, A. Lopez, and D. Gašević. *Tutorbot Corpus: Evidence of Human-Agent Verbal Alignment in Second Language Learner Dialogues*. International Educational Data Mining Society, July 2019. Publication Title: International Educational Data Mining Society.
- [27] N. Suk. The effects of extensive reading on reading comprehension, reading rate, and vocabulary acquisition. *Reading Research Quarterly*, 52(1):73–89, 2017.
- [28] G. Yali. L2 Vocabulary Acquisition Through Reading—Incidental Learning and Intentional Learning. *Chinese Journal of Applied Linguistics*, 33:pp. 74–93, 2010.
- [29] C. Y. Yeung and J. Lee. Personalized Text Retrieval for Learners of Chinese as a Foreign Language. In *Proc. of COLING*, pages 3448–3455, Aug. 2018.

Table A.1: Texts that are frequently on the efficient frontiers of the top 30 learners.

The number of times that the text appeared on the efficient frontiers	The first 30 words of the text. (All texts were taken from the Brown corpus.)
14	In the imagination of the nineteenth century the Greek tragedians and Shakespeare stand side by side , their affinity transcending all the immense contrarities of historical circumstance , religious belief
13	The Theatre-by-the-Sea , Matunuck , presents “ King Of Hearts ” by Jean Kerr and Eleanor Brooke . Directed by Michael Murray ; ; settings by William David Roberts .
12	She describes , first , the imaginary reaction of a foreigner puzzled by this “ unseasonable exultation ” ; ; he is answered by a confused , honest Englishman .
11	Into Washington on President-elect John F. Kennedy’s Convair , the Caroline , winged Actor-Crooner Frank Sinatra and his close Hollywood pal , Cinemactor Peter Lawford , Jack Kennedy’s brother-in-law .
10	On the fringe of the amused throng of white onlookers stood a young woman of remarkable beauty and poise . She munched little ginger cakes called mulatto’s belly and kept
10	As autumn starts its annual sweep , few Americans and Canadians realize how fortunate they are in having the world’s finest fall coloring . Spectacular displays of this sort are
10	Broadway the unoriginals To write a play , the dramatist once needed an idea plus the imagination , the knowledge of life and the craft to develop it . Nowadays

APPENDIX

A. EXAMPLES OF EFFICIENT TEXTS

Tab. A.1 lists the texts that are frequently on the efficient frontiers of the top 30 learners. The example texts were taken from the Brown corpus, which is available in the Natural Language Toolkit (NLTK, <https://www.nltk.org/>).

B. DATASET AND CODE

The dataset used in this study was published in [4]. We plan to release our code at <http://yoehara.com/> or <http://readability.jp/>. The dataset by [4] is also available for download by following the links of these websites. Further information about this paper may be found in the following website: <https://rebrand.ly/edm223info>.

C. DISCUSSION

The fact that the top 30 learners were chosen in **Tab. A.1** does not mean that, in general, our method cannot support less-skilled learners. In this case, our experiments assume a situation where the test-takers in the dataset by [4] were to conduct extensive reading on the texts taken from the Brown corpus. What matters is this combination: the Brown corpus is simply too difficult for less-skilled learners on [4], and incidental learning does not occur for them. If we use a text set easier than Brown corpus, incidental learning is more likely to occur even for low-ability learners, and the proposed method may be effective.

Identifying Explanations Within Student-Tutor Chat Logs

Ethan Prihar
Worcester Polytechnic Institute
100 Institute Road
Worcester, Massachusetts
ebprihar@wpi.edu

Alexander Moore
Worcester Polytechnic Institute
100 Institute Road
Worcester, Massachusetts
ammoore@wpi.edu

Neil Heffernan
Worcester Polytechnic Institute
100 Institute Road
Worcester, Massachusetts
nth@wpi.edu

ABSTRACT

To improve student learning outcomes within online learning platforms, struggling students are often provided with on-demand supplemental instructional content. Recently, services like Yup (yup.com) and UPchieve (upchieve.org) have begun to offer on-demand live tutoring sessions with qualified educators, but the availability of tutors and the cost associated with hiring them prevents many students from having access to live support. To help struggling students and offset the inequities intrinsic to high-cost services, we are attempting to develop a process that uses large language representation models to algorithmically identify relevant support messages from these chat logs, and distribute them to all students struggling with the same content. In an empirical evaluation of our methodology we were able to identify messages from tutors to students struggling with middle school mathematics problems that qualified as explanations of the content. However, when we distributed these explanations to students outside of the tutoring sessions, they had an overall negative effect on the students' learning. Moving forward, we want to be able to identify messages that will promote equity and have a positive impact on students.

Keywords

Large Language Representation Models, On-Demand Tutoring, Online Learning Platforms

1. INTRODUCTION

Middle school mathematics students have been shown to benefit from on-demand support when struggling within online learning platforms [8]. These supports require time and expertise to create, which can impede the platform's ability to provide support at scale. Additionally, when attempting to personalize students' online educational experience, it is essential to have multiple supports available for the same content. There has been notable success when crowdsourcing these supports from the teachers that use the platform

E. Prihar, A. Moore, and N. Heffernan. Identifying explanations within student-tutor chat logs. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 773–777, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852938>

[5, 6], but there are other opportunities to collect supports, including from live chat logs between students and tutors, which are an intuitive resource for generating support messages because they contain messages from a tutor explaining how to solve a problem to a student. Therefore, we propose the utilization of large language representation models to algorithmically identify support messages from live tutors that can be distributed to students at scale. At this point, we have created and evaluated a method for identifying relevant tutor messages from chat logs. However, while our method was able to identify messages that seemed equivalent to explanations already used in ASSISTments, the results of an empirical evaluation of these messages' ability to help students found that they negatively impacted learning. We are looking for critiques of our existing method and ways in which we can expand our method to account for more than just the semantic similarity between tutor messages and existing explanations in order to improve student learning outcomes.

2. METHODOLOGY

2.1 Data Collection

The student-tutor chat log data comes from the ASSISTments online learning platform [4] and UPchieve [2]. From June 9th, 2021 to November 5th, 2021, 82 students from 5 different classes had the opportunity to request help from a live UPchieve tutor within the ASSISTments learning platform. Over this time, students requested help 208 times and 8,817 messages were exchanged between students and tutors. In this work, we attempted to extract generalizable support from the live tutors' messages. Our approach relies on comparing large language model embeddings of the messages written by live tutors to the explanations already used as on-demand support within ASSISTments. Therefore, we also collected 16,130 existing explanations from the ASSISTments platform.

2.2 Identifying Generalizable Support

To identify generalizable support, we employ a four-step approach. The first step of our approach is to embed each tutor message and ASSISTments explanation using a large language representation model. We experimented with embeddings using three different pretrained large language models: BERT [3], SBERT [7], and MathBERT [9]. Due to the large number of features in these embeddings, the second step of our approach was to transform the embeddings using PCA [1] to mitigate over-fitting our subsequent supervised model. The third step was to train a logistic regression

to classify whether the embedded sample represented a tutor message or an ASSISTments explanation. The fourth step was to manually inspect the misclassified tutor messages and evaluate their generalizability and relevance. If they were deemed relevant, then they were included in the empirical study within ASSISTments. To select the number of PCA components to use for each logistic regression, a separate logistic regression was fit on 1 through n components, in order of decreasing significance, where n is the total number of components. After each logistic regression was fit, the total number of tutor messages classified as ASSISTments explanations was calculated and plotted. The plot revealed an "elbow", at which point additional PCA components had diminished effects on the accuracy of the logistic regression. Therefore, the number of components at the elbow of the graph was selected as the correct number of components to mitigate over-fitting.

2.3 Empirical Study

After candidate tutor messages were identified using the process described in Section 2.2, a randomized controlled experiment was performed within ASSISTments, in which an assignment consisting of six mathematics problems was given to middle school students. Only teachers that felt the problems were appropriate for their class allowed their students to participate in the experiment. For the 1st, 3rd, and 5th problems, students were randomized between receiving the identified tutor messages, or just the answer, as support upon their request. For the 2nd, 4th, and 6th problems, students were given a nearly identical problem with the same format and knowledge prerequisites as the previous problem. Within the assignment, the order that the students received the three pairs of problems was also randomized to eliminate bias from completing the problems in a particular order. The students' success on the 2nd, 4th, and 6th problems was used to evaluate the quality of the tutoring for each of the previous problems respectively. To determine the effectiveness of the tutor messages, an intent to treat analysis was performed in which Welch's t -tests [10] were used to compare the next-problem correctness of students that could have received the tutor messages and students that could have received the answer when requesting support, and Cohen's d was used to determine the effect size of the treatment compared to the control.

3. RESULTS

3.1 Identifying Generalizable Supports

Applying the elbow sample selection method described in Section 2.2 for each of the three different BERT models resulted in the plots shown in Figure 1. Only the first 20 PCA components are plotted because the elbow is always in the first 20 components. Based on Figure 1, MathBERT was the most accurate model and only misclassified eight tutor messages as explanations at the elbow, which is about 0.16% of all the tutor messages. To further illustrate the difference in the effectiveness of each BERT model, Figure 2 shows a two-dimensional projection of the regression boundary when fit using n PCA components, where n is determined by the elbow plots in Figure 1. Again, MathBERT has the least misclassifications. For this reason, the misclassified tutor messages from the logistic regression fit using the first eight PCA components of the MathBERT embedding were used

as candidate tutor messages for the empirical study. Of the eight messages, six were relevant enough to be used as on-demand support for the problems for which they were written. The other two messages were written about an example problem devised by the tutor, and not the original problem the student was trying to solve.

3.2 Empirical Study

The six selected tutor messages were written during discussion with students requesting support on three problems in ASSISTments. Therefore, the messages for each problem were combined into one set of on-demand support for the problem. Figure 3 shows each of the problems and the on-demand support created for them. In total, 163 students participated in the experiment. 106, 97, and 111 students were used to evaluate the supports for Problems 1, 2, and 3 respectively. Overall, there was a small, statistically insignificant negative effect from offering students tutor messages for support ($d = -0.145$, $p = 0.068$). When evaluating the effectiveness of the tutor message based supports for each individual problem, it was revealed that for the first and third problem, there was an insignificant negative effect from receiving on demand tutoring ($d = -0.094$ and -0.069 respectively, $p = 0.489$ and 0.627 respectively), but for the second problem, there was a medium-sized significant negative effect ($d = 0.406$, $p = 0.005$).

4. CONCLUSION

Although we were able to identify tutor messages that offered explanations to students on how to solve the problem they were struggling with, only 75% of the messages selected by our methodology were deemed relevant by human selection, and in an empirical study, these messages had a negative effect on real students' learning. This implies that there is more nuance to how helpful a message will be aside from how similar it is semantically to existing support messages. While this seems obvious, it is difficult to approach algorithmically filtering the tutor messages in a way that takes into account their relevance to the problem and how likely they are to benefit student learning outcomes. Moving forward, research efforts may compare tutor messages to the relevant problem, either in an embedding space or simply through phrase matching. This could help exclude tutor messages which were not relevant to the original problem but did get misclassified as supports in the embedding space. To address the negative impact that the selected tutor messages had on students' learning, we could attempt to use the students' responses in the student-tutor chat logs to identify only the tutor messages followed by student messages with positive sentiment, for example, "Thank you! I understand now". Ideally, by filtering the tutor messages this way, we would only include tutor messages that the student was happy to receive, which implies these messages would be more likely to help other students. Additionally, a metric to quantify the benefit to student learning outcomes given a problem and support would benefit our analyses of relevant tutoring logs. There are many factors to consider when improving student learning outcomes and we look forward to investigating these potential directions.

5. ACKNOWLEDGMENTS

We would like to thank NSF (e.g., 2118725, 2118904, 1950683, 1917808, 1931523, 1940236, 1917713, 1903304, 1822830, 1759229,

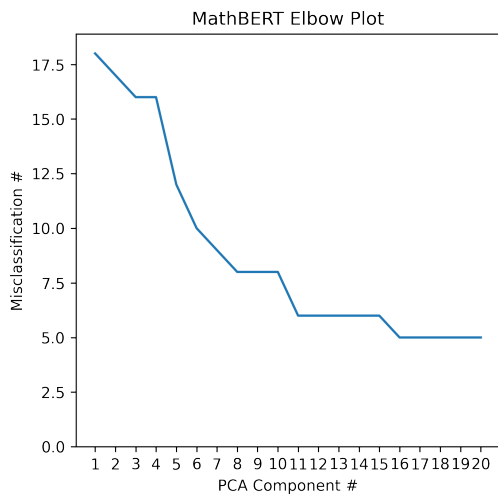
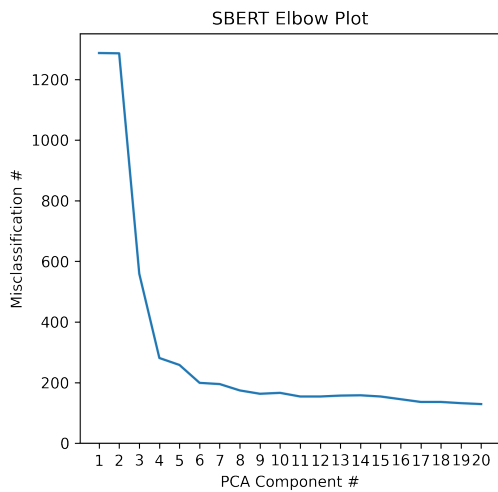
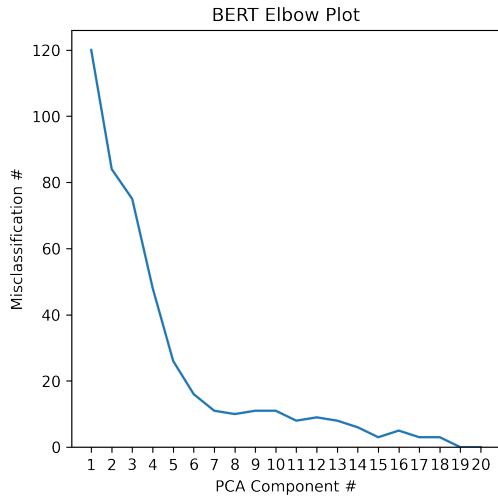


Figure 1: The elbow plot of the number of tutoring log messages misclassified as ASSISTments supports as a function of PCA components using embeddings from BERT (top), SBERT (middle), and MathBERT(bottom).

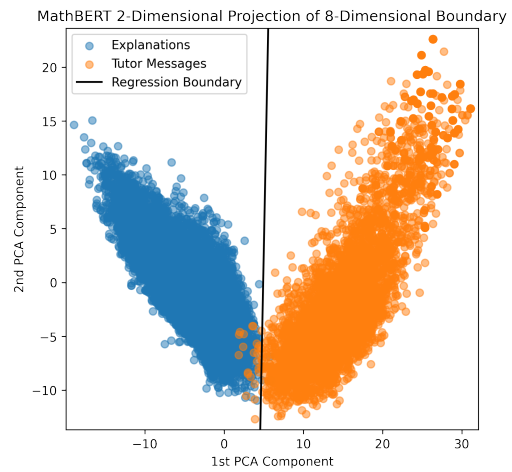
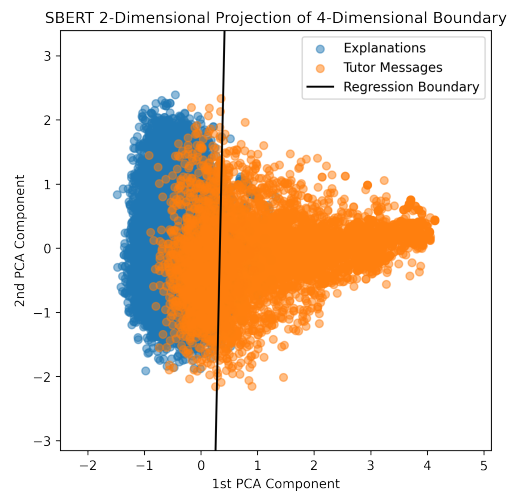
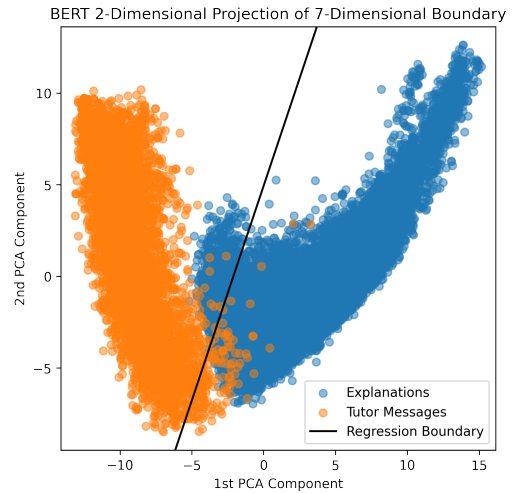


Figure 2: The data and regression boundary, projected to two dimensions, when using n PCA components, where n is determined by the elbow plots in Figure 1, to predict whether or not a message is from a live tutor or an ASSISTments explanation using BERT (top), SBERT (middle), and MathBERT(bottom) embeddings.

Problem ID: **PRABTHUB** [Comment on this problem](#)

What is the slope of the line graphed below?

slope = (units up or down) / (units to the left or right)

[Comment on this hint](#)

Here are some good resources to supplement your learning:

<https://www.khanacademy.org/math/algebra/x2f8bb11595b61c86:linear-equations-graphs/x2f8bb11595b61c86:slope/v/introduction-to-slope>

<https://www.khanacademy.org/math/algebra/x2f8bb11595b61c86:linear-equations-graphs/x2f8bb11595b61c86:slope/v/slope-of-a-line>

[Comment on this hint](#)

Type your answer below (mathematical expression):

33% ?

Submit Answer
Show answer

Problem ID: **PRABTHVF** [Comment on this problem](#)

$|10 \cdot 9 \cdot 5| - 10$

We can think of a number (say, -22, or 57) as having two features:

- the magnitude of the number (which we can think of as how far away the number is from 0) and the sign of the number (whether the number is less than 0 or greater than 0).

So, the magnitude of -22 is 22 and the sign is negative. Similarly, the magnitude of 57 is 57, and the sign is positive.

The absolute value of a number is only its magnitude.

For example, the absolute value of -22 is 22, and the absolute value of 57 is 57.

[Comment on this hint](#)

Type your answer below (mathematical expression):

67% ?

Submit Answer
Show answer

Problem ID: **PRABTHVK** [Comment on this problem](#)

The body of a 154-pound person contains approximately 2×10^{-1} milligrams of gold and 6×10^1 milligrams of aluminum. Based on this information, the number of milligrams of aluminum in the body is how many times the number of milligrams of gold in the body?

We need to divide the amount of aluminum by the amount of gold to find out how many times the amount of gold is equal to the amount of aluminum.

[Comment on this hint](#)

When we divide 6 by 2, of course we get 3. And when we divide 10^1 by 10^{-1} , we subtract the exponents.

The rule is, when you multiply two numbers with exponents that have the same base, you add the exponents. And when you divide two numbers with exponents that have the same base, you subtract the exponents.

[Comment on this hint](#)

Type your answer below (mathematical expression):

33% ?

Submit Answer
Show answer

Figure 3: The three problems related to the selected tutor messages and the messages themselves as they would be seen by the student using ASSISTments.

1724889, 1636782, 1535428), IES (e.g., R305N210049, R305D210031, R305A170137, R305A170243, R305A180401, R305A120125), GAANN (e.g., P200A180088 P200A150306), EIR (U411B190024), ONR (N00014-18-1-2768) and Schmidt Futures.

6. REFERENCES

- [1] H. Abdi and L. J. Williams. Principal component analysis. *Wiley interdisciplinary reviews: computational statistics*, 2(4):433–459, 2010.
- [2] J. Bryant, L.-K. Chen, E. Dorn, and S. Hall. School-system priorities in the age of coronavirus. *McKinsey & Company: Washington, DC, USA*, 2020.
- [3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [4] N. T. Heffernan and C. L. Heffernan. The assistments ecosystem: Building a platform that brings scientists and teachers together for minimally invasive research on human learning and teaching. *International Journal of Artificial Intelligence in Education*, 24(4):470–497, 2014.
- [5] T. Patikorn and N. T. Heffernan. Effectiveness of crowd-sourcing on-demand assistance from teachers in online learning platforms. In *Proceedings of the Seventh ACM Conference on Learning@ Scale*, pages 115–124, 2020.
- [6] E. Prihar, T. Patikorn, A. Botelho, A. Sales, and N. Heffernan. Toward personalizing students’ education with crowdsourced tutoring. In *Proceedings of the Eighth ACM Conference on Learning@ Scale*, pages 37–45, 2021.
- [7] N. Reimers and I. Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.
- [8] J. Roschelle, M. Feng, R. F. Murphy, and C. A. Mason. Online mathematics homework increases student achievement. *AERA open*, 2(4):2332858416673968, 2016.
- [9] J. T. Shen, M. Yamashita, E. Prihar, N. Heffernan, X. Wu, B. Graff, and D. Lee. Mathbert: A pre-trained language model for general nlp tasks in mathematics education. *arXiv preprint arXiv:2106.07340*, 2021.
- [10] B. L. Welch. The generalization of ‘student’s’ problem when several different population variances are involved. *Biometrika*, 34(1-2):28–35, 1947.

Detecting When a Learner Requires Assistance with Programming and Delivering a Useful Hint

Marcus Messer
King's College London
marcus.messer@kcl.ac.uk

ABSTRACT

Over the last few years, computer science class sizes have increased, resulting in tutors providing more support to struggling students, and instructors having less time per-student in larger classes. Universities typically assign multiple tutors to lab sessions, especially introductory programming courses, to maximise the help available to students during their sessions. However, using multiple tutors does not help struggling students outside of official sessions. The lack of support outside official settings is especially the case for online courses and remote learning. To help resolve student frustration from not being able to get support when they need it, we propose a tool that can detect when a student is struggling with their programming task and give them a hint that gets them closer to their goal.

Keywords

computer science education, computer programs, sequence mining, learning behaviours, feedback

1. INTRODUCTION

Over the last few years, the number of students enrolled in computer science courses has increased [12] and self-directed learning is becoming more prevalent. As class sizes grow, it becomes more difficult for course leaders and assistants to assist in programming labs. This difficulty is because it takes time to solve and explain various programming issues, and instructors have less time per-student in larger classes. Multiple tutors are frequently assigned to a single lab to maximise the availability of expert assistance. However, not all students will receive assistance during their scheduled lab sessions in some circumstances, particularly approaching deadlines, when multiple requests for help occur in a single lab session. Even if a tutor is available, many students do not seek assistance when struggling to solve a problem [15]. However, if a tutor offers a student support, they will usually accept it.

M. Messer. Detecting when a learner requires assistance with programming and delivering a useful hint. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 778–781, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852958>

Students have limited access to expert support outside of regular classes and office hours. The limited access is especially true for online courses, distance learning and students undertaking self-directed learning. The limited expert help in and outside official settings may lead to prolonged frustration, one of the leading causes of students dropping out [20].

A student could seek help for numerous programming areas, including compiler errors, logical errors, code style and best practice. Detecting code style issues and best practices are solved using various static code analysis tools, such as FindBugs and CheckStyle. FindBugs detects bad practices, performance, correctness, and “dodgy code” using pre-defined bug patterns, including common issues for novice programmers. Examples of which includes the comparison of String objects using “== or !=” and issues that cause `IndexOutOfBoundsException` [2]. CheckStyle detects various code style issues, including incorrect naming conventions and code design [7].

Some of the most challenging errors to solve are compiler errors, examples of which include “missing return statement”, “method call: parameter type mismatch”, and “method call targeting the wrong type” [16]. Understanding compiler error messages and correcting them is difficult and time-consuming for programmers of any degree of experience, especially novices [22]. Poorly written error messages could lead to a student not solving a problem for hours or days without help. Even with skilled help, an error could take several minutes to correct [13]. While compiler errors can be challenging to solve, logical errors are considerably more challenging to correct because they usually necessitate understanding the problem’s context and the student’s written work.

The proposed research will focus on detecting when a student requires assistance with compiler, logic and style errors and providing a meaningful hint on how to solve the error.

2. BACKGROUND & RELATED WORK

Previous work on detecting a struggling student has focused on other aspects than the source code. Spacco et al. used measures based on time and number of submissions to detect *flailing* students [21]. Rodrigo and Baker investigated detecting the aggregate frustration over many labs, using compilation based measures [20].

The compilation measures that Rodrigo and Baker implemented were the average time between compilations, the total number of compilations, the total number of errors, and Jadud’s Error Quotient (EQ) [20]. The EQ indicates how well a student can handle syntax errors while programming. It looks at successive pairs of compile events to see if both result in an error and if the error type is the same [9] (refined in [10]).

Recent research has investigated how EQ could be improved. Watson and Godwin created the Watwin algorithm, using consecutive compiler events to calculate a score; however, they also included checks for whether the error is on the same line and penalties for the time taken to solve each compilation pairing [23]. In addition, Becker developed the repeated error density (RED), which calculates a score based on repeated error strings, accounting for the lengths of repeated error strings in a sequence and assigning higher penalties for higher repeated error density [4].

While programming, students are given different types of feedback, the most common of which is next-step hints. After a student is stuck in a state and requires help to progress, a next-step hint will guide them to a more complete and correct version of the code [1].

There have been many implementations of next-step hints in different programming environments. Obermüller et al. have recently implemented next-step hints into Scratch. They first select a set of suitable candidate solutions using an automated test suite, then find the best matching candidate solution as the target solution. After they have the students and the target solution, they determine differences in the Abstract Syntax Tree (AST) to synthesise the hints [17]. However, this solution requires abundant candidate solutions.

There have been multiple attempts to provide hints in vast and sparsely populated spaces. Paaßen et al. produced the Continuous Hint Factory (CHF), which uses a supervised machine learning approach to provide a hint to the student. CHF uses the edit distance of the student’s current state and traces data of past students who have visited the same state in similar states to choose the edit with the lowest error as the hint [18].

While Paaßen et al.’s approach still require some trace data, Rivers and Koedinger use at least one reference solution and a test method to produce a solution space to find the next-step hint. They use a path construction algorithm to find the edit path from the current state to the solution, then finds the closest correct state within the space to find the edit that will be the basis of the next-step hint [19].

The previous approaches to next-step hint generation require trace data or a reference solution, but what if the student is the first to attempt such a task? Efremov et al. have used a reinforcement learning-based neural network hint policy to enable next-step hints for students attempting a programming task for the very first time [8].

3. PROBLEM STATEMENT

As Computer Science courses grow, the need for expert help increases; this is especially the case for online courses, remote learning and approaching coursework deadlines. Although having more tutors in regular sessions allows for more help, increasing the support offered in and out of the classroom would be beneficial.

Our research will investigate the following research questions:

- RQ1** How can traces of student code changes be used to detect struggling students?
- RQ2** After detecting a struggling student, when is the most effective time to give them a hint?
- RQ3** Can we generate next-step hints that focus on syntactic and logical errors comparable to the support offered by human tutors in these areas?

4. PROPOSED METHOD

The first phase of this project will determine whether or not a student is struggling. We will use Blackbox, a large dataset of novice programming data collected over the last eight years [6], to look for trends that could signal a struggling student. Blackbox is a large scale repository of novice programmers’ activity, including the length of programming sessions, compilation event history, and editing behaviour [6]. The editing behaviour includes a history of source code changes, allowing researchers to step through each change of a student’s code iteratively. We will explore patterns that could identify struggling students using the source code and compiler histories.

We will use existing research to detect at-risk students using compiled histories and invocations, such as the “Watwin” algorithm [23] and Repeated Error Density [5]. In conjunction with the compiler history, we will investigate the source code patterns that indicate that a student is struggling. We will use data mining approaches to derive developers’ behaviour patterns, similar to Kinnewbrew’s research into learners’ behaviour. Kinnewbrew et al. used interaction trace data from Betty’s Brain, which introduced students to a science topic (climate change). They abstracted and labelled different activities into different categories to analyse the data using differential sequence mining, identifying differentially frequent patterns across two groups of sequences [11].

We will investigate using interaction traces of source code to detect struggling students. Source code traces will allow us to examine how the student develops and any patterns in their programming and debugging practices that could indicate that they are struggling. Using source code traces could detect more struggling students than just compiler histories, as source code traces enable us to analyse more than just compiler related issues. An example pattern of a struggling student could be commenting out large chunks of code, potentially followed by uncommenting small portions to figure out which part of the code is causing their problem, which could signify that the student has trouble debugging and locating the problem. Another example may be the number of revisions made to a single code block in a given timeframe, suggesting that they are having trouble solving

a logical or syntactic mistake, depending on the compilation result.

We will use the patterns we found to detect when a student is struggling and give them a hint.

In order to give students a next-step hint, we will integrate the ideas of the Continuous Hint Factory (CHF) [18] and Evremov et al.'s research on using reinforcement learning to generate next-step hints [8]. We will use CHF as our primary method of producing hints, using Blackbox as the basis of potential candidate solutions. Blackbox contains data from opted-in users of BlueJ [14]. BlueJ is an educational IDE used worldwide by many different institutions at many different education levels. Accompanying BlueJ is a widely used textbook [3] that provides example projects and tasks, of which there are multiple solutions within Blackbox. However, there will be cases where a student is programming a task for the first time, such as new coursework. These cases are where Evremov et al.'s approach comes into effect, the ability to still provide next-step hints for the very first attempts of new programming tasks without the requirement of producing candidate or model solutions.

We will use Blackbox to learn patterns of struggling students and for elements of hint generation, as Blackbox contains both compile history and source code changes. To implement the ideas discussed in this section, we will create a plugin for BlueJ, which will run continuously in the background analysing the students' compile history and source code traces, comparing them to any patterns we have learnt from Blackbox. The plugin will include a "human-in-the-loop" process, enabling the user to give feedback to the system if the detection of their struggle was correct or if the hint helped solve their issue.

The following are the proposed steps that the system will follow:

1. While the student develops and compiles, the system will analyse the source code and compile histories for defined patterns.
2. If a pattern is detected, ask the user if they would like a hint.
 - (a) If yes, give the student a hint and ask them to rate the quality and usefulness of the hint.
 - (b) If no, log that they did not want a hint. If enough students give feedback that they do not want a hint for this specific pattern, flag it for investigation.
3. If the same pattern on the same part of the code is detected, repeat steps 2a and 2b, but with a more specific hint.
4. Repeat the steps above while the student develops.

We will conduct a series of trials to evaluate our proposed tool to determine if our system correctly detects struggling students and if the hints delivered are meaningful. Our trials could include asking students of various skills levels to

complete a set of programming tasks of increasing difficulty, with and without our proposed tool. After they complete the programming tasks, we could interview the students to determine if they thought they were offered a hint at the correct time and if the hint given was helpful. In addition to interviewing students, we may ask tutors for their feedback on whether the suggested method provided timely assistance and whether the hints assisted students in learning.

5. RISKS OF NEXT-STEP HINTS

This project aims to generate hints that will trigger a learning effect. We will have to hypothesise what area of the programming task the student is struggling with in order to generate a hint that will increase the students' understanding in that area. Using patterns of student development behaviour with next-step hint techniques could indicate which area the student is struggling with and give them a hint that increases their understanding and help them continue with their programming task.

While employing next-step hints might help students learn, there is a risk that using incrementally more specific hints can harm learning by giving students answers without teaching them how to improve. Students who abuse the hint system in this way exacerbate the risk of them not learning how to improve. We will investigate different methods to decrease the possibility of abuse in traditional hint systems, including providing a hint only when the system recognises that the student is having difficulty. However, this may not eliminate the risk, as some students will learn how to manipulate erroneous detections in order to obtain consecutive suggestions that will lead to the correct answer.

These risks lead to the pedagogical question: *Should the system give the student the answer if they have fundamental misunderstandings?* While undertaking this project, we will aim to minimise the risks discussed in this section and maximise the learning effect of the generated hints.

6. PROPOSED CONTRIBUTIONS

The proposed contributions of the above research will include:

- A method that can detect a struggling student using compiler error history and source code changes.
- Investigate when is the optimal time to hint after detecting a struggling student.
- A tool for detecting struggling students and giving them an appropriate next-step hint at the right time.

These contributions will benefit the Computer Science Education community by offering a tool to support students inside and outside formal education settings and alleviate some growing pressure on tutors of large classes.

7. CONCLUSION

We proposed the development of a tool that can detect when a programmer, mainly a novice programmer, is struggling and provide them with a next-step hint. We will look into how various pattern recognition techniques and past work on

metrics can be used to identify a struggling student based on compiler and code change histories.

8. REFERENCES

- [1] V. Aleven, I. Roll, B. M. McLaren, and K. R. Koedinger. Help helps, but only so much: Research on help seeking with intelligent tutoring systems. *Int J Artif Intell Educ*, 26:205–223, 2016.
- [2] N. Ayewah, D. Hovemeyer, D. J. Morgenthaler, J. Penix, and W. Pugh. Using static analysis to find bugs. *IEEE Software*, 25:22–29, 2008.
- [3] D. J. Barnes and M. Kölling. *Objects First With Java - A Practical Introduction Using BlueJ*. Pearson, sixth edition, 2016.
- [4] B. A. Becker. A new metric to quantify repeated compiler errors for novice programmers. *Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE*, 11-13-July-2016:296–301, 7 2016.
- [5] B. A. Becker. A new metric to quantify repeated compiler errors for novice programmers. *Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE*, 11-13-July-2016:296–301, 7 2016.
- [6] N. C. C. Brown, M. Kölling, D. McCall, and I. Utting. Blackbox: A large scale repository of novice programmers’ activity. *Proceedings of the 45th ACM technical symposium on Computer science education*, 2014.
- [7] O. Burn. Checkstyle. <http://checkstyle.sourceforge.net/>, 2003.
- [8] A. Efremov, A. Ghosh, and A. Singla. Zero-shot learning of hint policy via reinforcement learning and program synthesis. *International Conference on Educational Data Mining (EDM)*, 2020.
- [9] M. C. Jadud. Methods and tools for exploring novice compilation behaviour. *ICER 2006 - Proceedings of the 2nd International Computing Education Research Workshop*, 2006:73–84, 2006.
- [10] M. C. Jadud, M. Mercedes, T. Rodrigo, E. Tabanao, M. Beatriz, and E. Lahoz. Analyzing online protocols to characterize novice java programmers. *Philippine Journal of Science*, 138:177–190, 2009.
- [11] J. S. Kinnebrew, K. M. Loretz, and G. Biswas. A contextualized, differential sequence mining method to derive students’ learning behavior patterns. *Journal of Educational Data Mining*, 5:190–219, 2013.
- [12] S. Krusche, L. M. Reimer, B. Bruegge, and N. von Frankenberg. An interactive learning method to engage students in modeling. *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: Software Engineering Education and Training*, 20, 2020.
- [13] M. Kölling. The design of an object-oriented environment and language for teaching. pages 145–146, 1999.
- [14] M. Kölling, B. Quig, A. Patterson, and J. Rosenberg. The bluej system and its pedagogy. <http://dx.doi.org/10.1076/csed.13.4.249.17496>, 21:249–268, 2010.
- [15] F. Lee. When the going gets tough, do the tough ask for help? help seeking and power motivation in organizations. *Organizational Behavior and Human Decision Processes*, 72:336–363, 12 1997.
- [16] D. McCall and M. Kölling. A new look at novice programmer errors. *ACM Transactions on Computing Education (TOCE)*, 19, 7 2019.
- [17] F. Obermüller, U. Heuer, and G. Fraser. Guiding next-step hint generation using automated tests. *Proceedings of the 26th ACM Conference on Innovation and Technology in Computer Science Education V. 1*, 2021.
- [18] B. Paaßen, B. Hammer, T. W. Price, T. Barnes, S. Gross, and N. Pinkwart. The continuous hint factory - providing hints in vast and sparsely populated edit distance spaces. 8 2017.
- [19] K. Rivers and K. R. Koedinger. Data-driven hint generation in vast solution spaces: a self-improving python programming tutor. *International Journal of Artificial Intelligence in Education 27:1*, 27:37–64, 10 2015.
- [20] M. M. T. Rodrigo and R. S. J. Baker. Coarse-grained detection of student frustration in an introductory programming course. *ICER’09 - Proceedings of the 2009 ACM Workshop on International Computing Education Research*, pages 75–79, 2009.
- [21] J. Spacco, P. Denny, B. Richards, D. Babcock, D. Hovemeyer, J. Moscola, and R. Duvall. Analyzing student work patterns using programming exercise data. *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*.
- [22] V. J. Traver. On compiler error messages: What they say and what they mean. *Advances in Human-Computer Interaction*, 2010, 2010.
- [23] C. Watson, F. W. Li, and J. L. Godwin. Predicting performance in an introductory programming course by logging and analyzing student programming behavior. *Proceedings - 2013 IEEE 13th International Conference on Advanced Learning Technologies, ICALT 2013*, pages 319–323, 2013.

A Paraphrase Identification Approach in Paragraph length texts

Arwa Al saqaabi
Dr. Craig Stewart
Dr. Eleni Akrida
Prof. Alexandra Cristea

ABSTRACT

How to measure the semantic similarity of natural language is a fundamental issue in many tasks, such as paraphrase identification (PI) and plagiarism detection (PD) which are intended to solve major issues in education. There are many approaches that have been suggested, such as machine learning (ML) and deep learning (DL) methods. Unlike in prior research, where detecting paraphrases in short and sentence-level texts has been done, we focus on the not yet explored area of paraphrase detection in paragraphs. We consider that the meaning of a piece of text can be broken into more than one sentence, this is over and above the sentences as extracted from two benchmark datasets (Webis-CPC-11 and MSRP). TF-IDF, Bleu metric, N-gram overlap, and Word2vec are used as features, then SVM is invoked as a classifier. The contribution of this paper clearly indicates that, on a commonly used evaluation set, text at the length of a paragraph is more appropriate to consider than short or long text for ML and DL approaches. Additionally, our method outperforms the existing work done on the Webis-CPC-11 dataset.

Keywords

Natural Language Processing, Machine Learning, Deep Learning, Paraphrase Identification, Paragraph Length

1. INTRODUCTION

When students submit their work, institutions have to verify if the work is free of plagiarism. To overcome the limit of human abilities in terms of scalability (e.g. time to check and consistency) machine and deep learning techniques are applied for plagiarism and paraphrase detection tasks. Plagiarism is defined as using someone's written work without giving reference to the original source, or claiming the ideas taken from the work of others [22]. In some instances, the copying of many words from the original source, regardless of the provision of a reference, is also considered an act of plagiarism [4]. The modification of sentences in such a way that the original structure of the sentences, without acknowledgment, is used by the author, also falls in the category of plagiarism. According to Ehsan et al. [14], plagiarism detection methods are divided into two main categories, which are *intrinsic* plagiarism detection and *external* plagiarism detection methods. Intrinsic methods are implemented to detect the parts of the text that are inconsistent, while external methods can match suspicious passages in a text to

the source(s), detecting exact verbatim copying and paraphrased text [23].

Bhagat and Hovy [6] define paraphrasing as a means of conveying the same meaning, but with different sentence structure and wording. This definition clearly does not include exact verbatim reproduction as a case of paraphrasing. For our purposes, let us have two different texts, A and B. If the information, ϕ , which can be derived from A, can also be inferred from B, and vice versa, then A is a paraphrase of B (Equation 1): α represents a given domain or background knowledge [8].

$$(A \wedge \alpha | = \phi) \Leftrightarrow (B \wedge \alpha | = \phi), \text{ where } A \neq B \quad (1)$$

From this definition, it is obvious that paraphrase identification (PI) is implicitly part of plagiarism detection. (PD) Both PI and plagiarism detection have assumed a tremendous importance for academic institutions, researchers, and publishers concerned for the preservation of academic integrity [3]. PI is a method that aims to measure the degree of similarity between two given texts [11, 15]. PI also helps determine whether the two texts share the same meaning, which plays a vital role in natural language applications, such as plagiarism detection, summarisation of textual material, and machine translation. Semantic similarity is also used in several other activities, such as to retrieve information [21], answer questions [5] [31,1] and clustering [7].

Attempts to solve the problem of paraphrase identification in past studies were mainly focused on comparing words in sentences [28,29], phrases in sentences [2], or sentence to a sentence [12, 24]. These studies achieved robust results. However, comparing each sentence in the suspicious document (i.e., a students' assignment), to all sentences in the source documents, is not an efficient approach for long texts. Additionally, existing studies are ignoring the fact that sentence semantics could be distributed in a paragraph as a passage-level paraphrase type, which is more complex to recognise. Thus, we aim to develop a method for recognizing paraphrasing in paragraphs, henceforth called *passage-level paraphrasing*. This approach considers a paragraph as a basic unit, avoiding comparing all sentences of the documents as separate entities.

Existing work investigated paraphrasing that is accrued at sentence level [16, 30]. Additionally, prior works count exact and quasi-exact sentences as a paraphrased text [2, 25]. To the best of our knowledge, this study is the first to allow for passage-level paraphrasing (beyond sentence-only) text length. We have chosen to focus on passage-level paraphrasing, as we argue that it is a common and naturally occurring way to consider paraphrasing, more so than the previously studied sentence-level paraphrasing. In addition, we analyse how the text length affects machine learning (ML) model accuracy. For this purpose, we compare ML approaches that are mainly based on handcrafted features, as well as state-of-the-art deep learning sentence representation models, such as word2vec.

A. A. Saqaabi, C. Stewart, E. Akrida, and A. Cristea. A paraphrase identification approach in paragraph length texts. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 782–788, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852990>

For further research, constricting a new dataset is required to consider the text length and paraphrasing type. For purpose of this research, we provide the following definitions:

- a. *Sentence level paraphrasing*: the meaning of one sentence is paraphrased into exactly one other sentence (example: data in the MSRP dataset);
- b. *Passage level paraphrasing*: the semantics of a piece of text of multiple sentences are paraphrased into a potentially different number of sentences without a one-to-one mapping between sentence semantics (example: data in the Webis-CPC-11 dataset);
- c. *Sentence-length level*: represents a **short text** length, which is less than 50 words;
- d. *Paragraph-length level*: represents a **mid-text** length, which consists of about 100 words, this is the average length of a paragraph [20].
- e. *Passage-length level*: **long text** which consists from more than a paragraph (containing 150+ words).

Hence, our main research questions (RQ) are:

RQ1: How does the length of a piece of text affect the efficiency of the paraphrase identification approach used?

RQ2: What type of features are more effective for the problem of paraphrase identification on sentence - and passage level of paraphrasing, respectively?

RQ3: How effective are current state-of-the-art paraphrase identification methods for the problem of paraphrase identification in long text (paragraph) and (passage-level paraphrase)?

2. RELATED WORK

Machine learning (ML) and deep neural networks (DNN) attract the PI researchers and a lot of efforts have been devoted in this area. In these works, features are extracted by considering N-gram overlap features, metrics like Bleu, syntactic features, and semantic features from external knowledge such as WordNet or pretrained word embedding.

Cordeiro et al. [9, 10] implemented experiments that extract features from text, by applying a variety of metrics such as Bleu, edit distance which calculates how many character or word insertions, deletions, and replacements are required to change one string into the other, and Sim, Word Simple N-gram Overlap. They evaluated the experiments on the MSRP and the Knight and Marcu Corpus (KMC), where the paraphrased sentence is a shortened or summarized version of the original one. The Sim metric present the highest accuracy after removing the equal and quasi-equal samples from the dataset. For more investigations on the metrics' efficiency, they defined two types of paraphrasing, which are *symmetrical paraphrasing* (SP) and *asymmetrical paraphrasing* (AS). Symmetrical sentence pairs contain the same information, while in the asymmetrical paraphrasing, at least one sentence has more information. The result shows that the Sim metric is efficient for AS, while the Logisim metrics, based on the theory of exclusive lexical links between pairs of short text, is better for SP. Rather than implementing a specific threshold value to do binary classification, these metrics also were fed as text features extracted from the Webis-CPC-11 dataset, which has longer text samples, to a classifier such as SVM and k-nearest neighbours [8]. Regardless of the accuracy of these models, the Sim metric is suitable only for short texts, because of its demands on computing time.

Ferreira et al. [16] evaluated different ML algorithms, namely RBFNetwork, BayesNet, C4.5, and SMO on short text. They measured the lexical features from a Bag of Words (BOW), which breaks a text into all of its unique words and counts the number of times each word appears, syntactic and semantic features from the Resource Description Framework (RDF) based on dependence tree graphs. It mainly tackled two specific issues: sequences with the same meaning, but different terms, and the word-order problem. The results of the RBFNetwork and BayesNet algorithms outperform others, with accuracies of 75.13 and 74.08, respectively, both of which are measures of performance discussed in more detail in Section 3.3. Despite the fact that it did not improve overall outcomes, it significantly recognized the meaning of sentences that shared the same words but in a different order.

Wan et al. [29] designed an approach that considered 17 syntactic dependency features, to examine their effects on the accuracy of different machine learning algorithms, namely Naive Bayes learner, C4.5 decision tree, support vector machine, and K-nearest neighbour, to indicate dissimilarity between a pair of sentences. They claimed that dependency and N-gram features enhanced the classifier to recognise falsely paraphrased cases. In addition, avoiding lemmatisation was shown to keep the signs of differences in meaning and focus between sentences. However, more of the correctly paraphrased cases were identified as a negative, decreasing the overall accuracy of the approach. Moreover, they evaluated their experiment on a partial MSPC, because some cases led to stopping the parsing script. To leverage the limitations of this study, Ji and Eisenstein [18] considered the same features of Wan et al. [29] and implemented them on the whole MSRP corpus. Additionally, they developed a metric that computed the discriminability of features between sentences, called Term Frequency Kullback Leibler Divergence (TF-KLD). It counted the probabilities that appeared on paraphrased and non-paraphrased sentences, to re-weight features before factorisation, to obtain latent representations of the text. It clearly outperformed TF-IDF by 4% in accuracy and 1% in F1 score on MSRP. Moreover, they combined other features, such as unigram and bigram, overlapping fine-grained features, which raised the accuracy from 72.75 to 80.41. TF-KLD improved discriminatively distributional features while reducing others.

From another perspective, Vrublevskiy and Marchenko [28] extracted dependency tree, IDF and Bleu features from natural language. They concatenated word embedding with dependency tree features, to show that this combination can be useful to detecting paraphrase. However, this model did not outperform the state-of-the-art in that area [18].

Ji and Eisenstein [18] and Wan et al. [29] noted the need for more investigating on another dataset, also considering long text, such as a paragraph, where these studies examined only the MSPC corpus, where the maximum length of a sentence is 36 words [13]. In addition, using parse trees to solve problems restricted an approach to single sentences [19]. Also, BOW was unable to consider word order which is a vital textual feature in PI [30].

Nguyen et al. [24] developed an algorithm based on external knowledge and word embedding. It takes name entities (e.g., US) and rewrites them in words (e.g., United States). Additionally, they have applied the continuous bag of words CBOW and Skip-Gram models to extract interdependent features based on pre-trained word embedding. CBOW predicts a target word based on its context and Skip-Gram does the opposite, predicting context words according to the target word. As a part of the methodology, more features were also included that help to measure semantic relatedness based on external knowledge resources such as WordNet.

Features extracted from sentences with and without pre-processing. Then Support Vector Machine SVM is involved for classification task. It examined on MSRP, SemEval and P4PIN datasets achieving high accuracy 84.17, 83.73, and 95.22 respectively by considering features based on the external-knowledge resource. Unfortunately, they didn't report other evaluation metrics like F1 score, precision, and recall.

The above-mentioned work considered words or sentences as a mean unit of calculating semantic similarity in text segments. In addition, methods like SVM and BOW fall short of delivering a high-quality solution for extracting the semantics meaning of natural language. Particularly, SVM works better when integrated with deep learning models for recognizing paraphrased sentences [32].

Several academics have recently used deep neural networks to model sentence pairs to leverage the shortness of non-deep learning methods. Vrbanec and Meštrović [26, 27] conducted two studies on different models of sentence semantic representation such as word2vec, Glove and Fast-Text. The experiments were done on three datasets namely MSRP, Webis-CPC-11 and C&S. Because of the pairs of sentences that are semantically unrelated and very similar lexically, no specific model outperforms others on all datasets, however, USE provides high accuracy and F1 score. They also compare similarity metrics with two types of word embedding deep learning models: word2vec and Fast-Text [27].

Kenter and De Rijke [19] focused on word2vec and Glove semantic sentence representation. They extract word level and short sentence level features by word alignment and word embedding beside the saliency weighted semantic graph. This approach is corpus-based hence no use of external knowledge source is needed. It measures the semantic similarity on sentence level ignoring the importance of the word order in PI downstream task. Although applying word alignment to extract syntactic and semantic relations between words of the sentence and feeding them into SVM model as features showed the significance result on PI task, it needs to be examined on paragraph level. The authors show that classifiers which trained to predict semantic similarities between short texts can benefit from saliency-weighted semantic networks. In addition, concatenating of pre-trained word embedding models obtain better scoring than WordNet-based approaches.

Although these studies were done on more than one of public paraphrase dataset, they did not take into account the verity of numbers of words on each sample, nor the type of paraphrase have been done on different datasets. More importantly, they consider typical samples as paraphrased cases that do not state the paraphrase definition.

To overcome the limitation with sentence representation models that are based on a unidirectional encoder, Devlin et al. [12] proposed Bidirectional Encoder Representation from Transformers (BERT), which uses a masked language model and next sentence prediction, and is fine-tuned with one additional output layer. BERT has been demonstrated to achieve state-of-the-art outcomes on a wider array of sentence-level and token-level NLP tasks. Specifically in the PI task, it evaluated on MSRP with 89.30 percent of accuracy. This high accuracy raises the machine prediction accuracy to be closer to human performance.

As we see, BERT as a transformer learning model outperforms other sentence representation models because it generates a context vector representation that can discriminate word meaning in different contexts rather than giving the same weight for each word wherever it occurs [12].

Deep learning techniques have attracted a lot of attention in different research fields regarding to its impressive performance. In the PI field, researchers employ deep learning models to detect semantic similarity mainly in short text. In a way to show the efficiency of deep learning models over machine learning on PI task. Hunt et al. [17] compared the accuracy of two machine learning models with three different deep neural network models. Results illustrate that all DL models' accuracy outperforms LR and SVM models. The lowest accuracy is obtained by Siamese NN (~62) while the best accuracy is (~82) from LSTM RNN on the PI task.

3. METHODOLOGY

Prior research has investigated the efficiency of pre-processing techniques such as removing stop words and word lemmatizing [29], similarity metrics such as cosine, soft cosine and Euclidean [27] and using pre-trained word embedding models [26] on the PI downstream task. Here we examine how the length of text affects the model's accuracy in determining the appropriate number of words that could provide enough semantic information for machine models. Specifically, do short texts (sentences), mid length texts (paragraphs), or long texts (passages and paragraphs), provide sufficient semantic detail for the models? What type of features are most appropriate to use when attempting to identify paraphrases in sentence or paragraph texts? More importantly goal is that how to measure semantic meaning in paragraph length level to enhance other fields such as plagiarism detection, summarization, and text matching. Additionally, how the state-of-the-art autoregressive method advance the PI task?

3.1 Dataset

3.1.1 Microsoft Paraphrase Corpus (MSRP)

Dolan et.al. [13] presented separated sets of sentence pairs for training and evaluation. MSRP contains 4076 pairs of short text for train and 1725 for evaluation, taken from news sources on the internet. Human reviewers have then determined if each pair has a semantic equivalence. Each sentence pair is then labelled by 0 or 1, which represent negative and positive labels respectively.

3.1.2 Webis Crowd Paraphrase Corpus 2011 (Webis-CPC-11)

Burrows et.al. [8] provided 7859 possible text paraphrases pairs by Mechanical Turk crowdsourcing. The corpus consists of 4067 acceptable paraphrased pairs (meaning that one piece of text is a paraphrase of the other) and 3792 non-paraphrased pairs.

Most of the existing PI experiments are done on MSRP, which contains sentence-level paraphrases, so the existing results measure sentence similarity, whereas the most common and natural paraphrase is at the passage-level as seen in Webis-CPC-11. Another point of comparison between these corpora is the length of text which is vital for our study. The maximum sentence length in MSRP is 36 words while in the Webis-CPC-11 it is about a thousand words. This variety of document lengths in the Webis-CPC-11 dataset enables us to study how the text length can affect the ML model's results and to determine the best length of text that could be applied for ML and DL models.

3.2 Features Extracted

Based on the purpose of this study, we select the most important features in the PI task that represent text into numeric values. Each work discussed in Section 2 has implemented at least one feature of TF-IDF, Bleu, dependency tree, N-gram overlap or Word2vec [9, 10, 17, 18, 19, 24, 26, 27, 29]. However, most of these works

are done on the MSRP dataset which represents sentence level paraphrase, whereas we present a study on passage level paraphrase as seen in the Webis-CPC- 11 dataset.

4. EXPERIMENT

As we aim to study how the length of a text can affect the accuracy of using a specific or a combination of features in ML and DL models, we use MSRP as it focuses on short text length and Webis-CPC-11 as it contains a variety of text length samples. Therefore, we clean, then divide the samples from Webis-CPC-11 into three subsets based on the text length after removing the empty samples containing no text. Firstly, we remove identical sample texts from the Webis-CPC-11 dataset to satisfy the requirements of the paraphrase definition mentioned in the introduction, see equation (1). So, given a text pair (text 1, text 2), text 1 must be different from text 2 but carrying the same meaning. We call the resulting new dataset Webis-CPC-21, following the trend of the original dataset naming where 11 refers to 2011, the year of creation. However, we perform our experiments on the Webis corpus both with and without these identical samples to compare our results to state-of-the-art literature where possible. Secondly, we split the Webis-CPC-21 into three sub corpora: short text, where the maximum length of samples is 50 words; mid text represents the paragraph length (51-150 words) as the average length of a paragraph in English consists of 100 words [20]; long text containing samples of 151-500 words. We keep MSRP in its original form, as its samples consist of short text length (less than 40 words). Each dataset has numbers of negative and positive labelled samples. Positive and negative label refer to pair of text that are paraphrased and non-paraphrased respectively.

Table 1. The experiment's result, bold font represents the highest accuracy and f1 in each feature

Dataset	Bleu		TF-IDF		Sen2vec		Ngram_overlap		All Features	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
Webis-CPC-11	57	72	87	82	64	63	56	68	66	65
Webis-CPC-21	77	87	78	87	80	88	83	90	79	88
Short text	77	82	78	87	79	88	77	82	79	88
Mid length	81	89	83	90	84	91	85	91	85	91
Long text	73	84	73	84	75	89	78	86	75	85
MSRP	67	80	71	80	72	82	69	81	71	81

4.4 Result and Discussion

Our result in table 1 outperforms the baseline system of Webis-CPC-11 dataset which is the main dataset for this study, thanks to its variety of text length and the paraphrase type that is applied on its samples. Our accuracy and F1 are 80 and 88, respectively, which outperform baseline results by 3% in terms of F1 score on Webis-CPC-21 for Sen2Vec feature set. While the result on WebisCPC-11 is more by 3% for accuracy and less by 3% F1 with implementing one feature which is TF-IDF, rather than using 10 different metrics as the baseline system did. In addition, the efficiency of the ML and DL models could be improved when the length of the text is neither short nor long. Thus, the features engineers have to consider also must include the text length when extracting these features from text segments.

Our results on each feature on MSRP are worse than the ones achieved on the short text category, which potentially indicates that the good results we have achieved on Webis-CPC-21 are because the length of the samples in MSRP is even shorter than the short text category of Webis-CPC-21. Various previous studies

4.1 Pre-Processing

The data cleaning process includes removing irrelevant punctuation and stop words, which are commonly used words such as 'a', 'in' and 'the'. There isn't a single list of stop words that applies to every NLP task; we use the stop words list constructed by NLTK (Natural Language Toolkit) in python. Additionally, the process involves converting all letters to a lower case, then lemmatizing each word.

4.2 Feature sets

Since we're interested in how different features perform on different categories of text length, we carry out experiments per feature (TF_IDF, Bleu metric, sen2vec and N-gram overlap) and with combinations on the original dataset, modified dataset, and the sub-datasets that consist of different text sample lengths.

4.3 Baselines

The ground truth on Webis-CPC-11 dataset is determined by [8], where the Precision (P) is 81, Accuracy (Acc) is 84, and Recall (R) is 90. Although F1 is not reported, we calculate it by equation (2), which results in a value of 85 for F1. These results are obtained in [8] by feeding 10 different metrics as features to the k-nearest neighbour machine learning algorithm.

$$F1 = \frac{2PR}{P+R} \quad (2)$$

applied a pre-trained word2vec with cosine similarity or soft cosine on MSRP [27]. In this work, we convert each piece of text into one vector by summing up all word vectors in the text. This means we consider the semantic substance of the text to represent the overall text meaning. This clearly brings high accuracy and F1 results on Webis-CPC-21. In general, the Sen2vec yields the highest F1 score on most of the categories and surprisingly N-gram overlap performs much better than expected.

5. CONCLUSION

In this study, we answered RQ1 and RQ2 by investigating how the length of a text effects the model results in terms of measuring the semantic similarity of two different texts and which features work better with sentence, paragraph, and passage length levels. From the present experiment's results, we show that paragraph length level can convey the semantic meaning of natural language text better than sentence or passage length levels.

Based on the results, we plan to build a new passage-level- paraphrasing dataset that consist of a paragraph-length level to

achieve our contribution. Then involve the state-of-the-art transformer models to detect paraphrasing.

6. REFERENCES

- [1] Aouicha, M. B., Taieb, M. A. H., and Hamadou, A. B. (2018). SISR: System for integrating semantic relatedness and similarity measures. *Soft Computing*, 22(6), 1855-1879
- [2] Arase, Y., and Tsujii, J. (2021). Transfer fine-tuning of BERT with phrasal paraphrases. *Computer Speech & Language*, 66, 101164.
- [3] Bach, N. X., Le Minh, N., and Shimazu, A. (2014). Exploiting discourse information to identify paraphrases. *Expert Systems with Applications*, 41(6), 2832-2841
- [4] Bär, D., Zesch, T., and Gurevych, I. (2012). Text reuse detection using a composition of text similarity measures. *Proceedings of COLING 2012*, 167-184.
- [5] Barrón-Cedeño, A., Vila, M., Martí, M. A., and Rosso, P. (2013). Plagiarism meets paraphrasing: Insights for the next generation in automatic plagiarism detection. *Computational Linguistics*, 39(4), 917-947.
- [6] Bhagat, R., and Hovy, E. (2013). What is a paraphrase?. *Computational Linguistics*, 39(3), 463472.
- [7] Bollegala, D., Matsuo, Y., and Ishizuka, M. (2007, April). An integrated approach to measuring semantic similarity between words using information available on the web. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference* (pp. 340-347).
- [8] Burrows, S., Potthast, M., and Stein, B. (2013). Paraphrase acquisition via crowdsourcing and machine learning. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 4(3), 1-21.
- [9] Cordeiro, J., Dias, G., and Brazdil, P. (2007, March) a. A metric for paraphrase detection. In *2007 International Multi-Conference on Computing in the Global Information Technology (ICCGI'07)* (pp. 7-7). IEEE.
- [10] Cordeiro, J., Dias, G., and Brazdil, P. (2007) b. New functions for unsupervised asymmetrical paraphrase detection. *Journal of Software*, 2(4), 12-23.
- [11] Das, D., and Smith, N. A. (2009, August). Paraphrase identification as probabilistic quasi-synchronous recognition. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP* (pp. 468-476)
- [12] Devlin, J., Chang, M. W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
- [13] Dolan, W., Quirk, C., Brockett, C., and Dolan, B. (2004). Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources.
- [14] Ehsan, N., Shakery, A., and Tompa, F. W. (2019). Cross-lingual text alignment for fine-grained plagiarism detection. *Journal of Information Science*, 45(4), 443-459
- [15] Fernando, S., and Stevenson, M. (2008, March). A semantic similarity approach to paraphrase detection. In *Proceedings of the 11th annual research colloquium of the UK special interest group for computational linguistics* (pp. 45-52).
- [16] Ferreira, R., Cavalcanti, G. D., Freitas, F., Lins, R. D., Simske, S. J., and Riss, M. (2018). Combining sentence similarities measures to identify paraphrases. *Computer Speech & Language*, 47, 59-73.
- [17] Hunt, E., Janamsetty, R., Kinares, C., Koh, C., Sanchez, A., Zhan, F., ... and Oh, P. (2019, November). Machine learning models for paraphrase identification and its applications on plagiarism detection. In *2019 IEEE International Conference on Big Knowledge (ICBK)* (pp. 97-104). IEEE
- [18] Ji, Y., and Eisenstein, J. (2013, October). Discriminative improvements to distributional sentence similarity. In *Proceedings of the 2013 conference on empirical methods in natural language processing* (pp. 891-896).
- [19] Kenter, T., and De Rijke, M. (2015, October). Short text similarity with word embeddings. In *Proceedings of the 24th ACM international on conference on information and knowledge management* (pp. 1411-1420).
- [20] Larock MH, Tressler JC, and Lewis CE (1980) Mastering effective English. Copp Clark Pitman, Mississauga
- [21] Li, H., and Xu, J. (2014). Semantic matching in search. *Foundations and Trends in Information retrieval*, 7(5), 343-469.
- [22] Maurer, H. A., Kappe, F., and Zaka, B. (2006). Plagiarism-A survey. *J. Univers. Comput. Sci.*, 12(8), 1050-1084.
- [23] Muangprathub, J., Kajornkasirat, S., and Wanichsombat, A. (2021). Document plagiarism detection using a new concept similarity in formal concept analysis. *Journal of Applied Mathematics*, 2021.
- [24] Nguyen, H. T., Duong, P. H., and Cambria, E. (2019). Learning short-text semantic similarity with word embeddings and external knowledge sources. *Knowledge-Based Systems*, 182, 104842.
- [25] UI-Qayyum, Z., and Altaf, W. (2012). Paraphrase identification using semantic heuristic features. *Research Journal of Applied Sciences, Engineering and Technology*, 4(22), 4894-4904
- [26] Vrbanec, T., and Meštrović, A. (2020). Corpus-based paraphrase detection experiments and review. *Information*, 11(5), 241
- [27] Vrbanec, T., and Meštrović, A. (2021). Relevance of Similarity Measures Usage for Paraphrase Detection
- [28] Vrublevskiy, V., and Marchenko, O. (2020, November). Paraphrase Identification Using Dependency Tree and Word Embeddings. In *2020 IEEE 2nd International Conference on Advanced Trends in Information Theory (ATIT)* (pp. 372-375). IEEE.
- [29] Wan, S., Dras, M., Dale, R., and Paris, C. (2006, November). Using dependency-based features to take the 'parafarce' out of paraphrase. In *Proceedings of the Australasian language technology workshop 2006* (pp. 131-138)
- [30] Wang, X., Li, C., Zheng, Z., and Xu, B. (2018, July). Paraphrase recognition via combination of neural classifier and keywords. In *2018 International Joint Conference on Neural Networks (IJCNN)*(pp. 1-8). IEEE.

- [31] Yang, Y., Yih, W. T., and Meek, C. (2015, September). Wikiqa: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 conference on empirical methods in natural language processing* (pp. 2013-2018).
- [32] Yin, W., Schütze, H., Xiang, B., and Zhou, B. (2016). Abcnn: Attention-based convolutional neural network for modeling sentence pairs. *Transactions of the Association for Computational Linguistics*, 4, 259-272

PRESENTATION LETTER

Arwa Al saqaabi
1st year PhD student at
Durham university
arwa.alsqaabi@durham.ac.uk

PhD supervisor(s):

1- Dr Craig Stewart
<https://www.durham.ac.uk/staff/craig-d-stewart/>

2- Dr Eleni Akrida
www.durham.ac.uk/staff/eleni-akrida/

3- Professor Alexandra Cristea
<https://www.durham.ac.uk/staff/alexandra-i-cristea/>

Title: A Paraphrase Identification Approach in Paragraph length texts

I mainly work on Paraphrase identification downstream task as it important for other tasks such as plagiarism detection. These tasks could be solved by machine learning and deep learning approaches. The existing work done on sentence-length level and sentence-level paraphrasing while I focused on paragraph text length and passage-level paraphrasing.

In this year (1st year of my PhD), I implemented a study on how the length of text impact the model's results and submitted as a short paper for this conference. The experiment's results clearly show that the paragraph length level provides semantic meaning better than sentence or passage length levels. To expand my research and contribution, I plan to construct a dataset considering the text length and paraphrasing type to be in paragraph length level and passage-level paraphrasing as the existing dataset represent sentence-level paraphrasing.

I am seeking behind your recommendation, comment, opinion, and advice such on how to expand my research and What tool or approach I have to consider.

Investigating learners' Cognitive Engagement in Python Programming using ICAP framework

Daevesh Singh
Indian Institute of Technology Bombay
daeveshsingh@iitb.ac.in

Ramkumar Rajendran
Indian Institute of Technology Bombay
ramkumar.rajendran@iitb.ac.in

ABSTRACT

Introductory programming courses suffer a fate of high failure. Also, research has shown a significant relationship between engagement and academic success. Therefore, reliable and real-time engagement measures could help identify students who need help and provide them with personalised instruction. The present methods of measuring engagement rely primarily on self-reports that have reliability issues due to self-bias and poor recall and, at the same time, do not provide real-time and high granularity data. Interestingly, the online environments allow for real-time capturing of fine-grained interactions, which could be used to measure students' engagement and overcome the issues of self-report measures. The focus of our work will be cognitive engagement, which is a less explored dimension of engagement. To achieve this, we have developed an online learning environment called PyGuru for teaching-learning of Python. We propose to collect learner interaction data, classify these actions into different levels of cognitive-engagement, and study the impact of these different levels on their learning. We present the initial work done in this direction regarding the system developed and the data collected. We intend to seek advice on the validity and reliability of our approach to measuring cognitive engagement.

Keywords

Programming, Cognitive Engagement, PyGuru

1. INTRODUCTION

Research has demonstrated a significant relationship between engagement and educational outcomes like persistence, completion and achievement [7, 9, 18]. While student engagement is crucial to any learning experience, it is particularly important for domains like computer programming since introductory programming courses suffer a fate of high failure and dropouts and engagement as a construct can help in addressing this [2, 21].

D. Singh and R. Rajendran. Investigating learners' cognitive engagement in Python programming using ICAP framework. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 789–794, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852960>

The construct “engagement” has multiple unique conceptualisations. Kuh (2007) defines engagement as “participation in educationally effective practices, both inside and outside the classroom, which leads to a range of measurable outcomes” . Krause and Coates (2008) define it as “the extent to which students are engaging in activities that higher education research has shown to be linked with high-quality learning outcomes” [13, 14].

Engagement as a construct not only requires involvement but also feelings and sense-making [15] as a result, many theorists argue for a multidimensional definition of engagement. In literature, the various dimensions of engagement include academic, behavioural, emotional, affective, psychological, social, cognitive, and agentic dimensions [4, 8, 9, 11]. The focus of our work will be Cognitive Engagement(CE), because of two reasons. Firstly, as it has established research and a theoretical base that supports its significance in learning [3, 9, 10]. Secondly, CE has been less explored in the domain of computer programming [12]. Cognitive engagement is defined as the extent to which students are willing to invest in working on the task [5], and how long they persist [17, 19]. Similarly, as per Fredricks et al.(2004) “*cognitively engaged students would be invested in their learning, would seek to go beyond the requirements, and would relish challenge.*” [9].

Our research uses the framework developed by Chi & Wylie (2014). This framework classifies students' overt behaviours (like taking notes, asking questions etc.) into four different modes of engagement, namely interactive, constructive, active, and passive(ICAP)[3]. ICAP model, which was initially developed for classroom learning, has been extended to online learning as well [6, 16, 20, 22]. However, most of the existing studies in the online environment explore components such as video watching, DF, in pieces (i.e. they apply ICAP to only one of the online learning components like video watching, discussion forums, etc). This implies a lacuna of studies in a Computer-Based Learning Environment (CBLE) that study all these components together and hence, lacks an understanding of how students' behaviour in CBLE is related to their CE. Also, most studies that explore engagement in computer programming mainly focus on the motivational and behavioural aspect and not the cognitive aspect of engagement[12]. Hence, there is a need of analysing actions learners' perform in CBLE with all the existing components to better understand cognitive engagement and its impact on learning .

To fill the above gaps, we developed a CBLE called PyGuru for teaching-learning of Python programming that consists of four components: book-reader, video player, Integrated Development Environment(IDE), and discussion forum. This learning environment captures fine-grained data related to learning like highlights and annotations in the book-reader. Play, pause and seek behaviour and response to in-video questions in a video player. Views, comments, and likes in the discussion forum and their performance in IDE using test cases for each program.

In our research, we propose to use the interaction data generated to classify students' actions into different modes of cognitive engagement, namely Interactive, Constructive, Active and Passive (ICAP). We will then study the impact of these different engagement levels on students learning of computer programming. These results will help us develop models to measure learners' cognitive engagement and provide them feedback to prevent failure and dropouts.

In this paper, Section II presents the background and related works. The description of the learning environment and study design are provided in section III. Section IV offers a proposed solution, and the closing section contains questions for which we intend to seek advise.

2. BACKGROUND AND RELATED WORK

To analyze learners' interaction in the system, we will be using the ICAP framework developed by Chi Wylie (2014). This section consists of two subsections. The first one provides a brief introduction to the ICAP framework and the second subsection involves the related works that use the ICAP framework.

2.1 ICAP Framework

ICAP is a hierarchical framework that defines engagement in terms of students' overt behaviours and tries to distinguish these overt behaviours into four modes of engagement, namely interactive, constructive, active and passive[3].

They define passive mode as *"learners receiving information without overtly doing anything related to learning"* (Chi Wylie, 2014, p. 221). In this mode, learners simply attend to the information (without performing any actions like note-taking) and store it in episodic form rather than integrating it with prior knowledge.

Active engagement occurs when the learner's information acquisition is accompanied by certain physical or motoric actions that support their learning. This includes taking notes in the classroom, pausing and recap of the videos being watched or highlighting the text while reading.

The cognitive process involved during active engagement demands the activation of prior knowledge and integrating the new information into the existing one. Constructive engagement happens when the learners attempt to produce some artefacts using their prior knowledge and the information available in the environment. The characteristic feature of these artefacts is that they use information that goes beyond the available information. This includes "elaborating, comparing and contrasting, generalizing, reflecting on, and explaining how something works" (Chi Wylie, 2014, p. 228).

Interactive engagement occurs when learners engage in interaction, and during these interactions, both partners must be constructively engaged and there must be sufficient turn-taking.

2.2 Studies based on ICAP

Yogev et al. (2018) examined students' CE in reading material using Nota Bene annotation platform[22]. They firstly analyzed CE anchored in the text by manually labelling students' annotations and then developed an interactive decision tree to automate this process. They found that different sections of the text elicit different levels of CE. For instance, low CE corresponds to definitions provided in the text. They also developed a visualization tool to show the distribution of varying levels of CE over the reading material.

Acknowledging the absence of a framework for active viewing(active learning to describe students' behaviours while learning from video), Dodson et al. (2018) developed a framework to classify students' different video watching behaviours as per the ICAP framework which classified behaviours as passive(playing video content), active(replay, pause, seek specific information), constructive(taking notes, highlighting), and interactive (cooperating and collaborating with others) [6]. To fully support video-based learning, they used ViDeX in this study which supported a broader set of behaviours, as mentioned previously.

Wang et al. (2015) investigated students' contributions in the discussion forum participating in a MOOC[20]. They studied the relationship between the kind of posts students write and their learning gain. For this purpose, they considered each post as a sampling unit and coded each post into nine categories: active (note-taking, repeat, paraphrase), constructive (compare or connect, ask novel questions, provide justification or reason), and interactive (building on partner's contribution, acknowledging partner's contribution, defend or challenge). Their study shows a significant association between the discourse in the discussion forum and learning gains.

The study by Atapattu et al. (2019) tried to automate this process of classifying the post in the discussion forum into the active and constructive modes of engagement[1]. Their research included finding semantic similarities between each post and related course materials using cosine similarity. The posts significantly different from the learning materials were classified as constructive, and the more similar ones were labelled as active.

To summarise, we have presented how ICAP is used in book-reader, video-watching and discussion forums. These studies have influenced the design of our system PyGuru. Moreover, the results presented in these studies are promising and indicate a strong link between students' CE and their performance. It is, therefore, crucial to investigate how overall learning behaviour (actions in book reader, video player etc.) impact students' performance in Python programming. To understand this, we present PyGuru- a learning environment for learning Python programming capable of logging the user actions. The following section offers details about the system, study design, and the proposed solution.

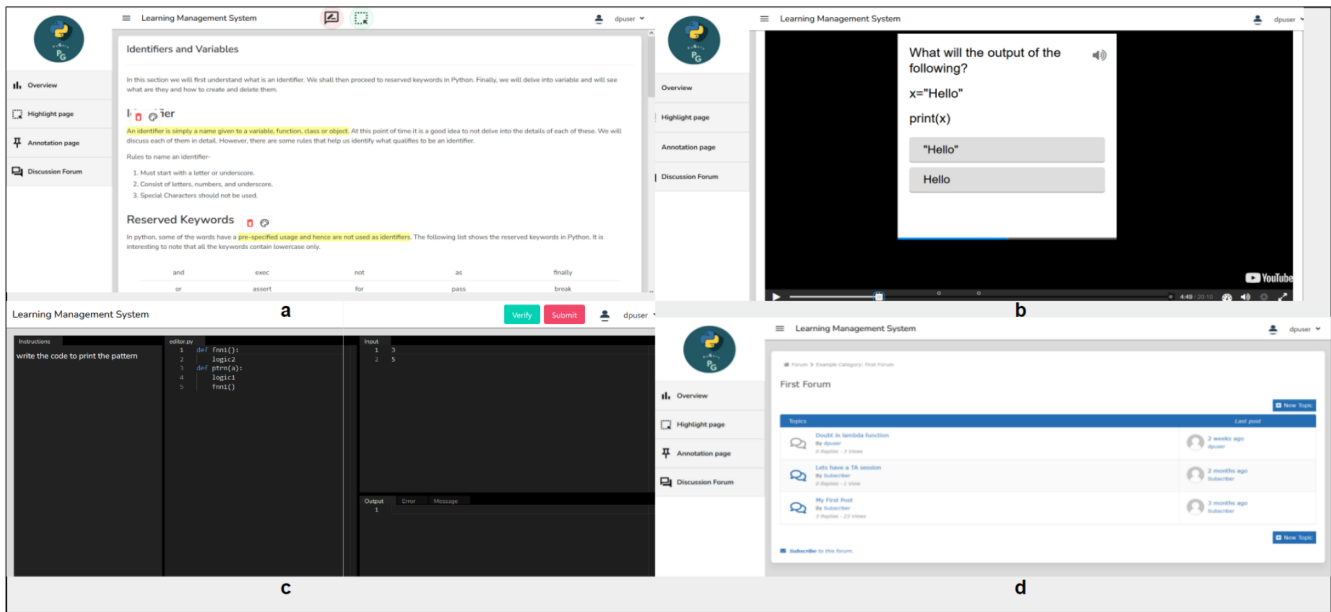


Figure 1: PyGuru Environment (a) Book Reader (b) Video Player (c) IDE (d) Discussion Forum

3. LEARNING ENVIRONMENT

PyGuru¹ is a computer-based learning environment for teaching and learning Python programming skills. PyGuru has four major components: 1) Book Reader 2) Video Player 3) Integrated Development Environment (IDE) 4) Discussion Forum.

A book reader (shown in Fig 1. a) in PyGuru allows readers to highlight and annotate the text. Highlight in the digital context consists of selecting a text and colouring it. The annotating feature in the learning environment comprises selecting a text, commenting on that text, and providing a tag to that text.

PyGuru has an interactive video watching platform (Fig. 1.b). The students can interact with the video using basic video player features like enhancing the speed of the video and performing other actions like play, pause and seek. More advanced interactive features are embedded into the system that allows the instructor to add questions within the video. The video automatically stops and waits for the learner's response.

We also have integrated an IDE (Fig 1. c) in PyGuru. Currently, the IDE evaluates the learners' code against the test cases. The "verify" button in the IDE allows learners' to check their program for errors and test cases before submitting. The IDE consist of four panels 1) Instruction Panel -To provide details about the problem or algorithm 2) Input Panel - To provide the test cases for the problem, and these test cases are tested against these inputs 3) Coding Panel - The coding panel is where the student is expected to write the code, and the instructor can present some partial codes as well 4) Output Panel - To display the output once the program is run and provide information about the number

¹<https://tinyurl.com/2p8h6zpw>

of test cases passed and failed.

PyGuru has a discussion Forum (Fig 1.d). The learners can use this discussion forum to put forth their queries which the instructor or their peers can then answer. They can also respond and hit like on the post made by their peers or instructor.

4. METHODOLOGY

The study is focused on investigating learners' cognitive engagement in a CBLE, designed for teaching and learning of Python programming. We decided to use ICAP framework not only to design the system but will also use it to analyze learners' interactions in PyGuru. The aim is to answer the following questions: 1.1 Is there a relationship between log data of student activity in CBLE and self-reported student engagement survey scores? 1.2 Is there a relationship between actions that are believed to represent higher cognitive engagement and learning gains? 1.3 How do learners' actions having different learning gains differ in a CBLE while learning computer programming?

4.1 Study Design

The study participants will be students from undergraduate colleges or diploma colleges located in developing countries (India and Malaysia). The duration of the study will be 3-5 weeks. During this time the learners will be interacting with the system to learn programming by reading content, watching videos, and coding on IDE. One week prior to the study, learners will be introduced to the system. Along with this a consent form, demographic survey and a pre-test will be administered. This pre-test will be designed by the instructor and on similar lines a post-test will be designed to measure learning gains. Student engagement will be measured using the 7-item activity-level student engagement survey (Henrie et al. 2016). From this survey,

we will only consider items corresponding to cognitive engagement. Log data from the PyGuru will be collected for all participants. The log data consists of the user id, session id, timestamp, page id, and other details like the action performed (e.g. highlighted or annotated) and the context information (the text highlighted).

4.2 Proposed Solution

We propose to investigate cognitive engagement in CBLE using log-data. At this point, there are two directions. One is to do a correlation analysis with the engagement scores obtained through engagement surveys and identify whether the data captured from logs could be used as a proxy for cognitive engagement. The second one is to use a data-driven approach to identify the characteristics of students with higher learning gains using descriptive and diagnostic analytics. Further, using the findings obtained from descriptive and diagnostic analytics to develop regression models to measure cognitive engagement.

5. ADVICE SOUGHT

Q1. Are the system's design and the data captured suitable for measuring engagement?

Q2. Since we plan to develop regression models for measuring engagement using learning gains. Is this proposed approach valid? Also, what are other ways to validate our measures other than self-report or teachers' ratings?

Q3. What are the general suggestions regarding the study design in terms of duration of the study, the data collected from the environment, etc.?

6. ACKNOWLEDGMENTS

I would like to thank Ashwin TS, Deepak Pathak and Ruma Pathan for useful discussions and their support. This paper is based upon work supported by the grant from Council of Scientific and Industrial Research(CSIR).

7. REFERENCES

- [1] T. Atapattu, M. Thilakaratne, R. Vivian, and K. Falkner. Detecting cognitive engagement using word embeddings within an online teacher professional development community. *Computers & Education*, 140:103594, 2019.
- [2] C. Beise, L. VanBrackle, M. Myers, and N. Chevli-Saroq. An examination of age, race, and sex as predictors of success in the first programming course. 2003.
- [3] M. T. Chi and R. Wylie. The icap framework: Linking cognitive engagement to active learning outcomes. *Educational psychologist*, 49(4):219–243, 2014.
- [4] S. Christenson, A. L. Reschly, C. Wylie, et al. *Handbook of research on student engagement*, volume 840. Springer, 2012.
- [5] L. Corno and E. B. Mandinach. The role of cognitive engagement in classroom learning and motivation. *Educational psychologist*, 18(2):88–108, 1983.
- [6] S. Dodson, I. Roll, M. Fong, D. Yoon, N. M. Harandi, and S. Fels. An active viewing framework for video-based learning. In *Proceedings of the fifth annual ACM conference on learning at scale*, pages 1–4, 2018.
- [7] J. D. Finn and J. Owings. *The adult lives of at-risk students the roles of attainment and engagement in high school—statistical analysis report*. DIANE Publishing, 2006.
- [8] J. D. Finn and K. S. Zimmer. Student engagement: What is it? why does it matter? In *Handbook of research on student engagement*, pages 97–131. Springer, 2012.
- [9] J. A. Fredricks, P. C. Blumenfeld, and A. H. Paris. School engagement: Potential of the concept, state of the evidence. *Review of educational research*, 74(1):59–109, 2004.
- [10] C. R. Henrie, R. Bodily, R. Larsen, and C. R. Graham. Exploring the potential of lms log data as a proxy measure of student engagement. *Journal of Computing in Higher Education*, 30(2):344–362, 2018.
- [11] C. R. Henrie, L. R. Halverson, and C. R. Graham. Measuring student engagement in technology-mediated learning: A review. *Computers & Education*, 90:36–53, 2015.
- [12] G. Kanaparan, R. Cullen, D. Mason, et al. Effect of self-efficacy and emotional engagement on introductory programming students. *Australasian Journal of Information Systems*, 23, 2019.
- [13] K.-L. Krause and H. Coates. Students' engagement in first-year university. *Assessment & Evaluation in Higher Education*, 33(5):493–505, 2008.
- [14] G. D. Kuh. What student engagement data tell us about college readiness. 2007.
- [15] S. J. Quaye, S. R. Harper, and S. L. Pendakur. *Student engagement in higher education: Theoretical perspectives and practical approaches for diverse populations*. Routledge, 2019.
- [16] M. Raković, Z. Marzouk, A. Liaqat, P. H. Winne, and J. C. Nesbit. Fine grained analysis of students' online discussion posts. *Computers & Education*, 157:103982, 2020.
- [17] J. C. Richardson and T. Newby. The role of students' cognitive engagement in online learning. *American Journal of Distance Education*, 20(1):23–37, 2006.
- [18] V. Trowler. Student engagement literature review. *The higher education academy*, 11(1):1–15, 2010.
- [19] C. O. Walker, B. A. Greene, and R. A. Mansell. Identification with academics, intrinsic/extrinsic motivation, and self-efficacy as predictors of cognitive engagement. *Learning and individual differences*, 16(1):1–12, 2006.
- [20] X. Wang, D. Yang, M. Wen, K. Koedinger, and C. P. Rosé. Investigating how student's cognitive behavior in mooc discussion forums affect learning gains. *International Educational Data Mining Society*, 2015.
- [21] C. Watson and F. W. Li. Failure rates in introductory programming revisited. In *Proceedings of the 2014 conference on Innovation & technology in computer science education*, pages 39–44, 2014.
- [22] E. Yogev, K. Gal, D. Karger, M. T. Facciotti, and M. Igo. Classifying and visualizing students' cognitive engagement in course readings. In *Proceedings of the Fifth Annual ACM Conference on Learning at Scale*, pages 1–10, 2018.

Presentation Letter

Name: Daevesh Singh

Institute: Indian Institute of Technology Bombay (IITB), Mumbai, India

Supervisor: [Prof. Ramkumar Rajendran](#)

Title: Measuring learner's Cognitive Engagement in Python Programming using log-data

Introduction: I am Daevesh Singh, a third year research scholar at the Educational Technology Department, IITB. My research area is educational data analytics. All my research work till now has been focused on using log-data from learning environments and their analysis. Some of my published works include-

1. Nishane, I., Sabanwar, V., Lakshmi, T. G., **Singh, D.**, & Rajendran, R. (2021, July). Learning about learners: Understanding learner behaviours in software conceptual design TELE. In *2021 International Conference on Advanced Learning Technologies (ICALT)* (pp. 297-301). IEEE.
2. Satavlekar, S., Nath, D., Priyadarshini, R., Prasad, P., **Singh, D.**, & Rajendran, R. (2021, July). Unraveling Learner Interaction Strategies in VeriSIM for Software Design Diagrams. In *2021 International Conference on Advanced Learning Technologies (ICALT)* (pp. 308-310). IEEE.
3. **SINGH, D.**, PATHAN, R., BANERJEE, G., & RAJENDRAN, R. From Hello to Bye-Bye: Churn Prediction in English Language Learning App.

About Thesis: My PhD thesis is focussed on measuring cognitive engagement using log-interaction data. The domain includes introductory programming. The study will involve diploma or undergraduate students that are learning Python as a part of their curriculum from the Indian Subcontinent (Mostly India and Malaysia) . For this purpose, we have developed a Computer Based Learning Environment called PyGuru. PyGuru consists of four components, namely the book-reader, video-player, Integrated Learning environment(IDE), and discussion Forum and the environment logs all the interactions made by the learner. We plan to use this data to come-up with meaningful measures of cognitive engagement.

EDM researchers:

1. Neil Hefferman - Neil's research involves computer-based learning environments and engagement.

2. Tanja Mitrovic - Tanja's work in active video watching could be helpful since my environment also has an interactive video player.
3. Sharon Hsiao - Sharon's work includes the use of learning analytics in the domain of computer programming.

Improving Automated Assessment and Feedback for Student Open-responses in Mathematics

Sami Baral
Worcester Polytechnic Institute
sbaral@wpi.edu

ABSTRACT

Advancements in online learning platforms have revolutionized education in multiple different ways, transforming the learning experiences and instructional practices. The development of natural language processing and machine learning methods have helped understand and process student languages, comprehend their learning state, and build automated supports for teachers. With this, there has been a growing body of research in developing automated methods to assess students' work both in mathematical and non-mathematical domains. These automated methods address questions of two categories; closed-ended (with limited correct answers) and open-ended (are often subjective and have multiple correct answers), where open-ended questions are mostly used by teachers to learn about their student's understanding of a particular concept. Manually assessing and providing feedback to these open-ended questions is often arduous and time-consuming for teachers. For this reason, there have been several works to understand student responses to these open-ended questions to automate the assessment and provide constructive feedback to students. In this research, we seek to improve such a prior method for assessment and feedback suggestions for student open-ended works in mathematics. For this, we present an error analysis of the prior method "SBERT-Canberra" for auto-scoring, explore various factors that contribute to the error of the method, and propose solutions to improve upon the method by addressing these error factors. We further intend to expand this approach by improving feedback suggestions for teachers to give to their students' open-ended work.

Keywords

Online Learning Platforms, Open-responses, Natural Language Processing, Machine Learning, Automated assessment, Mathematics

1. INTRODUCTION

S. Baral. Improving automated assessment and feedback for student open-responses in mathematics. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 795–798, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853036>

In the past decade, development in artificial intelligence and machine learning methods have led to advancements in online learning platforms, transforming learning experiences and teaching practices. From personalized learning to augmenting teaching processes through automated assessment methods [8, 2, 4, 14, 11, 1], the benefits of these platforms have been significant. With this, several prior works have leveraged machine learning methods and natural language processing-based techniques to automate the assessment of students' work, both in mathematical [10, 6] and non-mathematical domains [13, 15]. As these methods and models of student learning become deeply integrated into normal instructional and educational practices, it becomes increasingly important to understand the strengths and weaknesses in their application. Within this, it is important to not only identify areas where existing methods under-perform, but it is also important to develop methods to improve such models to alleviate risks to fairness.

In the domain of mathematics there have been several works to automate assessment and provide constructive feedback to students, to further increase the efficiency of teaching and help teachers guide their focus to students in need. These works address problems of two categories: close-ended and open-ended problems. For close-ended problems, that have a finite number of correct answers, these auto-scoring methods can apply simple matching techniques to compare the student answer with the list of correct answers and consistently achieve near-perfect accuracy. However, open-ended problems are subjective with multiple accepted correct answers and are mostly given in the form of natural language. For these types of responses teachers commonly assess students based on an explicit or implicit rubric that pinpoints key aspects that must be included in a student response to sufficiently demonstrate their understanding. In addition, these types of student responses in mathematics often are a combination of language, images, tables, or other mathematical expressions, equations, and terminologies, which poses a challenge in developing automated methods of assessment for these problems. Due to the numerous challenges that this poses to automated assessment, existing methods commonly apply natural language processing (NLP) to build a high-dimensional representation of student responses that is then combined with various machine learning approaches (e.g. [13, 15, 3, 5]).

In this paper, we observe one such prior works on automated assessment model of student open responses in mathemat-

ics based on sentence-level semantic representation of the student open responses: "SBERT-Canberra" method. With the goal of exploring the limitations and further improving the method for assessment, we discuss our prior study that applies an exploratory error analysis to identify the areas of improvement that may be addressed by future iterations of these methods. We further propose a simple solution to improve upon the SBERT-Canberra method for automated assessment by addressing one of these error factors, that is the presence of mathematical terms and expressions in student answers. Additionally we seek to explore and address other factors of error to further improve the SBERT-Canberra method for auto-scoring. Finally we also intend to improve and expand this work towards feedback recommendations for student open-responses in mathematics.

2. PRIOR WORK

For this research, we observe one of the prior works on automated assessment of student open-responses in mathematics: the "SBERT-Canberra" model. This method follows a simple similarity-ranking procedure to generate the score predictions based on Sentence-BERT (SBERT) [12]. When suggesting a score for a given student response, it first applies SBERT to generate a high-dimensional feature embedding that describes the response as a whole. The intuition behind this is to capture semantic and syntactic meaning within this embedding, such that similar responses would be mapped closer within the embedding space. The SBERT embedding for this new student response is then compared to SBERT embeddings corresponding to a pool of historic labeled student responses utilizing the canberra distance measure [9]. In the final step the score for the historic response corresponding to the smallest distance (i.e. the most similar response) is used as the score prediction. The intuition behind this method is that similar answers to the a problem would have the same score.

3. CURRENT WORKS

3.1 Error Analysis of Auto-Scoring Method

With the goal of exploring the limitations of the SBERT-Canberra approach in order to identify the areas where the model does well and where it may yet improve through future iteration, we conducted an exploratory error analysis of the method. The dataset for the analysis was collected during the pilot testing of a teacher-augmentation tool designed to aid the assessment of open-responses within the ASSISTments[7] online learning platform. This tool, called QUICK-Comments used the SBERT-Canberra model to predict the scores for student open-responses in mathematics. Toward the error analysis, we observe two regression models that observe absolute model error as a dependent variable. The absolute model error here is the absolute difference between the score predicted by the SBERT-Canberra model and the teacher assigned grade for a particular student answer.

3.1.1 Uni-level linear model

For the uni-level model we explore characteristics of student answers in the context of this modeling error. These answer-level features are composed of length of answer, average character per word in the answer, total nos. of numbers, and operators in the answer text, percentage of mathematical expression in the answer text, and presence of images.

The results of the error analysis are presented in Table 1. It is found that the uni-level linear model with student answer level features explains 38.6% of the variance of the outcome as given by r-squared. Out of the six student answer-level features, nearly all were found to be statistically reliable predictors of model error. However, only two of these variables: Equation Percent and Presence of Images were found to have more meaningful coefficient as compared to other features. This suggested that the presence of mathematical expressions and images(unsurprisingly) both correlate with higher prediction error.

3.1.2 Multi-level linear model

Similarly, we then apply a multi-level model to observe which of student-, problem-, and teacher-level identifiers most explains any observed modeling error. In regard to this, accounting for student, problem, and teacher identifiers each as random effects, we see that the inclusion of these level-2 factors explains some of the impact of the fixed effects (Table 1). It is worth noting that the level-2 variables account for 55.5% of the variance of the outcome. This suggests that a majority of the modeling error can be explained by the factors external to the student answers. Looking at the variance of the random effects, it can be seen that the problem level identifiers contribute most in terms of explaining the variance of the outcome.

3.2 Improving the Auto-Scoring method

With the results from the error analysis of the SBERT-Canberra method for auto-scoring, next we seek to improve this approach addressing for the factors that contributed to the modeling error. We know from the error analysis that one of the limitations of the SBERT-Canberra method in predicting the scores is in the presence of mathematical terms and equations in the student answer. To address this limitation, we propose the "Math Term Frequency" (MTF) model, drawing inspiration from assessment methods applied for close-ended problems. The goal of this method is to learn about the mathematical terms present in student answers and supplement this method to the previously developed SBERT-Canberra model through ensembling. For this, first we identify non-linguistic terms in students answers, and then identify the most frequently-occurring terms for each possible integer score to learn a kind of rubric. These most frequently occurring non-linguistic terms are then used to develop the features for this method. These features indicate whether a newly-observed student response contains any of the most frequent terms most commonly associated with each given score. They are finally used in a multinomial logistic regression (treating each score as an independent category), trained separately for each problem.

The score predictions from the MTF model are then ensembled with the SBERT-Canberra predictions using another logistic regression model, referred to as the SBERT-MTF model; to clarify, this ensemble regression model observes ten features corresponding to the probability estimates produced for each of the five possible scores for each of the two observed models. The goal of this is to combine the semantic representation captured by the SBERT method, while taking advantage of the non-linguistic term matching from the MTF method.

Table 1: The resulting model coefficients for the uni-level linear regression model and random and fixed effects of the multi-level linear model of absolute error for auto-scoring method.

	Uni-level Linear		Multi-level Linear	
	Variance	Std. Dev.	Variance	Std. Dev.
<i>Random Effects</i>				
Student	—	—	0.034	0.185
Problem	—	—	0.313	0.559
Teacher	—	—	0.048	0.851
	B	Std. Error	B	Std. Error
<i>Fixed Effects</i>				
Intercept	0.581***	0.017	0.772***	0.070
Answer Length	-0.008***	0.001	-0.009***	0.001
Avg. Word Length	-0.014***	0.003	-0.013**	0.003
Numbers Count	<0.001	<0.001	<0.001	<0.001
Operators Count	-0.006***	0.001	0.002	0.001
Equation Percent	0.443***	0.018	0.080***	0.022
Presence of Images	2.248***	0.021	1.858***	0.028
Answer Length X Images	-0.081***	0.004	—	—

*p < 0.05 **p < 0.01 ***p < 0.001

4. FUTURE WORKS

In this research, we have identified areas where more advanced methods of image processing and natural language processing (or math language processing), may lead to further improvements in the existing methods for automated assessment. We have proposed a simple solution to address the limitations of current scoring method in presence of non-linguistic terms in student answers. While this proposed solution is an initial step towards addressing mathematical terms in these NLP based methods, we intend to explore more advanced methods based on Mathematical language processing and MathBERT to address such issues in future.

Further, we believe this method can be extended to recommend feedback messages in addition to suggesting numeric scores. With this, the next steps in our research is to expand the existing methods in suggesting and generating directed feedback to these student answers. We believe that the proposed MTF method combined with SBERT-Canberra can be extended as a prediction task in predicting whether given student responses are similar or not. This could be further beneficial in finding similar answers to math open-ended questions and thus utilizing this in improving the feedback recommendation task.

While the current works are based on textual open-ended responses, there are other forms of open-ended responses in mathematics including drawn diagrams and graphs, hand written formulas and expression uploaded as images, and other forms of audio and video responses. We seek to expand this research to these other forms of student open-ended responses, and further study the feasibility in deploying these automated methods in a computer-based learning environment.

5. ACKNOWLEDGMENTS

We thank multiple NSF grants (e.g., 1917808, 1931523, 1940236, 1917713, 1903304, 1822830, 1759229, 1724889,

1636782, 1535428, 1440753, 1316736, 1252297, 1109483, & DRL-1031398), as well as the US Department of Education for three different funding lines; the Institute for Education Sciences (e.g., IES R305A170137, R305A170243, R305A180401, R305A120125, R305A180401, & R305C100024), the Graduate Assistance in Areas of National Need program (e.g., P200A180088 & P200A150306), and the EIR. We also thank the Office of Naval Research (N00014-18-1-2768) and finally Schmidt Futures we well as a second anonymous philanthropy.

6. REFERENCES

- [1] J. Burstein, C. Leacock, and R. Swartz. Automated evaluation of essays and short answers. 2001.
- [2] E. Chen, M. Heritage, and J. Lee. Identifying and monitoring students' learning needs with technology. *Journal of Education for Students Placed at Risk*, 10(3):309–332, 2005.
- [3] H. Chen and B. He. Automated essay scoring by maximizing human-machine agreement. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1741–1752, 2013.
- [4] L. Cutrone, M. Chang, et al. Auto-assessor: computerized assessment system for marking student's short-answers automatically. In *2011 IEEE International Conference on Technology for Education*, pages 81–88. IEEE, 2011.
- [5] S. Dikli. An overview of automated scoring of essays. *The Journal of Technology, Learning and Assessment*, 5(1), 2006.
- [6] J. A. Erickson, A. F. Botelho, S. McAteer, A. Varatharaj, and N. T. Heffernan. The automated grading of student open responses in mathematics. In *Proceedings of the Tenth International Conference on Learning Analytics & Knowledge*, pages 615–624, 2020.
- [7] N. T. Heffernan and C. L. Heffernan. The assistments ecosystem: Building a platform that brings scientists

- and teachers together for minimally invasive research on human learning and teaching. *International Journal of Artificial Intelligence in Education*, 24(4):470–497, 2014.
- [8] K. Holstein, G. Hong, M. Tegene, B. M. McLaren, and V. Aleven. The classroom as a dashboard: Co-designing wearable cognitive augmentation for k-12 teachers. In *Proceedings of the 8th international conference on learning Analytics and knowledge*, pages 79–88, 2018.
- [9] G. Jurman, S. Riccadonna, R. Visintainer, and C. Furlanello. Canberra distance on ranked lists. In *Proceedings of advances in ranking NIPS 09 workshop*, pages 22–27. Citeseer, 2009.
- [10] A. S. Lan, D. Vats, A. E. Waters, and R. G. Baraniuk. Mathematical language processing: Automatic grading and feedback for open response mathematical questions. In *Proceedings of the Second (2015) ACM Conference on Learning@ Scale*, pages 167–176, 2015.
- [11] M. Mohler and R. Mihalcea. Text-to-text semantic similarity for automatic short answer grading. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 567–575, 2009.
- [12] N. Reimers and I. Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks, 2019.
- [13] B. Riordan, A. Horbach, A. Cahill, T. Zesch, and C. Lee. Investigating neural architectures for short answer scoring. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 159–168, 2017.
- [14] L. Zhang, Y. Huang, X. Yang, S. Yu, and F. Zhuang. An automatic short-answer grading model for semi-open-ended questions. *Interactive learning environments*, 30(1):177–190, 2022.
- [15] S. Zhao, Y. Zhang, X. Xiong, A. Botelho, and N. Heffernan. A memory-augmented neural model for automated grading. In *Proceedings of the fourth (2017) ACM conference on learning@ scale*, pages 189–192, 2017.

Modeling Cognitive Load and Affect to Support Adaptive Online Learning

Minghao Cai

EdTeKLA Research Group, University of Alberta
Edmonton, Canada
minghaocai@ualberta.ca

Carrie Demmans Epp

EdTeKLA Research Group, University of Alberta
Edmonton, Canada
cdemmansepp@ualberta.ca

ABSTRACT

Online learning has been spreading with the increasing availability and diversity of digital resources. Understanding how students' cognitive load and affect changes when using learning technologies will help us decipher the learning process and understand student needs. In this research, we focus on modeling learner's cognitive load and affect using real-time physiological reactions. We explore what affect modeling contributes to the modeling of cognitive load, and how real-time cognitive load changes alongside learning activities. We want to further investigate if cognitive load modeling helps diagnose learner knowledge and facilitates improvement. We have designed two case studies: one where students are learning python with an e-learning system and another where they are practicing literacy skills with a web-based learning game. To collect learner data, we have implemented a sensing prototype consisting of an eye tracker and a wireless wristband.

Keywords

Student modeling, Cognitive load, Affect, Online learning.

1. INTRODUCTION

The cognitive processes that underpin information processing and knowledge construction in learning have gained increasing attention from educational researchers. Prior studies have found cognitive load has a complex relationship with learning performance [22]. Models like cognitive load theory (CLT) [24, 33] have been used to explain the learner's cognitive process. How student cognitive load reacts when dealing with an abundance of information is a key component of learning performance. While previous research has shown the impact of cognitive load on learning outcomes, the exploration of its real-time changes when learners use online educational technologies has not yet been fully conducted. Moreover, the diversification of how learning materials are presented and changes in the corresponding learning methods suggest a need to explore their impact on learner cognitive load.

Although CLT effectively represents learner processing, some have argued that it fails to fully explain learning performance [21, 24]. Recent research has suggested people's affective reactions might contribute to changes in their cognitive load during learning given the role that affective states seem to play in learning [10]. Neuroscience research has suggested that the brain mechanisms

M. Cai and C. D. Epp. Modeling cognitive load and affect to support adaptive online learning. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 799–804, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853020>

underlying affect and cognition are not fundamentally different [18]. The potential interactivity between affect and cognitive load [9, 11, 16] suggests that there may be value in jointly modelling these constructs.

Previous learner modeling studies using cognitive load mostly focused on supporting adaptation (e.g., feedback [6], problem selection [17]) to improve student performance. We argue that by including learner affect alongside their cognitive load, learner modeling has the potential to help improve learner's experiences while supporting effective learning.

In this thesis project, we explore the potential of modeling students' real-time cognitive load when using online educational tools to support adaptive learning experiences. We highlight a learner's affective reactions and their contributions to cognitive load, as one of the areas that has not been sufficiently considered in previous student modeling studies. Based on recent research in cognitive processes in learning, we believe a learner's cognitive load is affected by the difficulty of tasks and the sequence of learning units so it can change alongside the learning process.

To achieve our goal, user studies across different educational domains will be conducted. The online learning environments include one that supports learning to program in python and an edu-game for supporting literacy development. A sensing platform has been built to collect and synchronize different types of learner physiological data. Multimodal analysis will be conducted to detect learner cognitive load and affect. The modeling of cognitive load will be used for predicting the learner's knowledge in each domain.

2. RELATED WORK

2.1 Cognitive Load and Affect

CLT is widely used. Its application in educational design to enhance instruction has supported the learning of children, teens [35], and older adults [23, 38] in a wide variety of domains. CLT is grounded in an understanding of the human cognitive architecture that is supported by previous research on working memory models [2] and mental effort [31].

There are three types of cognitive load. Intrinsic load (IL) is the inherent difficulty of what needs to be learned and should be managed so it does not exceed the capacity of working memory. Extraneous load (EL) is unnecessary load that does not support learning [20, 34]. It is usually caused by specific learning activity or system designs. The third kind is germane load (GL) which refers to the mental resources devoted to acquiring and automating schemata.

CLT can help understand student's learning patterns and provide guidance for adaptive learning design. However, we expect that it will not be enough to model the cognitive load of students since recent neuroscience research has shown the interconnectedness of

affect and cognition despite earlier theories that posited the processes were separate [26]. Studies suggest that cognitive and affective behaviors have rich interactions and some argue that emotions arise from the same general cortical system that processes cognition [18]. These findings support the joint inclusion of affect and cognitive load within learner models.

While research on the connection of affect and learning has been performed for a long time [37], many studies have explored affect without considering its implication on cognitive load. For example, previous research on negative emotions argued students experiencing longer periods of boredom tended to have lower scores than those experiencing positive emotions [7]. However, the type of negative affect is important; students might have better learning performance when given erroneous examples even though those examples produced confusion and frustration [5].

While some argue the interrelatedness of affect and cognition, there is no consensus on how to incorporate affect into cognitive load modeling. Some argue affect could increase extraneous cognitive load because emotion regulation may add non-task-related processing that consumes extra cognitive resources [12]. Others have suggested that affect could be considered beneficial as it can foster motivation so that learners invest more cognitive effort [14]. Even negative emotions could motivate learners to turn to learning for shifting their attention away from their negative emotional states [4]. In this thesis project, we would like to recognize a student's affective reactions in the context of computer-supported online learning and seek to investigate how affect contributes to learner cognitive load in these settings.

2.2 Measurement of Cognitive Load & Affect

To achieve our goal, appropriate data collection and modeling approaches must be selected. Commonly used methods include subjective rating scales as well as task- and performance-based methods. According to Paas and van Merriënboer's work, cognitive load can be assessed using aspects of mental load, mental effort, and performance [25]. Subjective rating scales have been widely used and are grounded in a belief that people can reflect on their cognitive processes and report the amount of mental effort expended. The NASA task load index (NASA-TLX) is one instrument that attempts to capture this type of information [13]. However, it might not be able to measure unconscious and automatic processes. Task-based and performance-based methods have also been commonly used to measure cognitive load by measuring reaction time or accuracy on a secondary task. Although monitoring such tasks itself requires few cognitive resources, their use may interfere with the primary task when a reaction is necessary [8].

As the nature of questionnaires and performance evaluation implies, the above two methods rely on data collection after an experiment, which fails to support continuous monitoring. Brain-activity-based methods such as electroencephalography (EEG) have been used for identifying changes in cognitive load [1]. This measurement approach requires wearing complex instruments that are obtrusive to users. More recently, with advances in sensing technologies, physiological-data-based methods have presented the potential to address this issue. Eye-tracking technology is one of these sensors that can now be worn like a pair of glasses. This sensor provides information about pupillary response, which is considered a reliable source that enables the investigation of cognitive processes. One project found that pupil diameter changes indicated visual presentations that induced lower cognitive load in a math-education context [15]. More generally, increased cognitive load has been associated with increases in pupil diameter due to

decreasing parasympathetic activity in the peripheral nervous system [3].

In addition to estimating cognitive load through dynamic pupillary information, research has shown that user's gaze trajectory data can be reliable for quantitatively measuring reading behavior. For example, eye-gaze data has been used to infer user cognitive style in reading activities [28] and the impact of distractions on surgeons' intraoperative performance [32]. Such visual information is the key to investigating learner attention patterns and strategic processes.

In the context of affect measurement, three groups of tools are commonly used: psychological (i.e., self-reported), physiological, and behavioral [39]. While psychological methods often depend on respondents' ability to consciously process their affective responses, physiological methods allow researchers to capture non-conscious aspects. Physiological arousal data, such as Electrodermal activity (EDA) and skin temperature, are considered to provide robust signals for measuring affect [29].

In this thesis project, we incorporate questionnaires and physiological tracking including eye-tracking data, cardiovascular responses, and EDA data for cognitive load and affect measurement.

3. RESEARCH QUESTION & APPROACH

Considering the research problem mentioned above, the overall goal of this thesis is to *model students' cognitive load and affect across instructional domains*. To address this goal, case studies will be conducted to collect learner data, including their performance and reactions to learning activities. Models will then be developed using the collected data. Across this work we will answer the following questions.

- Q1 What does affect modeling contribute to the modeling of cognitive load in online learning settings?
- Q2 How does learner affect and cognitive load change alongside their learning when interacting with educational systems?
- Q3 Can affect and cognitive load help in identifying a learner's knowledge state and trajectories?

To answer these questions, we have designed and will conduct lab-based case studies. Two case studies will be performed with different educational technologies: one is used to teach undergraduate students how to program and the other helps students improve their reading comprehension.

3.1 What does affect modeling contribute to the modeling of cognitive load in online learning settings?

We would like to investigate the added benefit, if any, of jointly modeling affect and cognitive load so we can determine what affect contributes to learner cognitive load modeling.

Considering the dynamic, sequential nature of cognitive load, we will use hidden Markov models (HMM) to model learner's cognitive load. An HMM approach demands that the system has observable evidence that suggests the value of a hidden state. We will use the physiological reactions as observable signals to estimate the hidden cognitive load state.

We will compare the performance of estimating intrinsic and extraneous cognitive load using physiological features, i.e., pupil dilation (PD) and heart rate against the performance of estimating

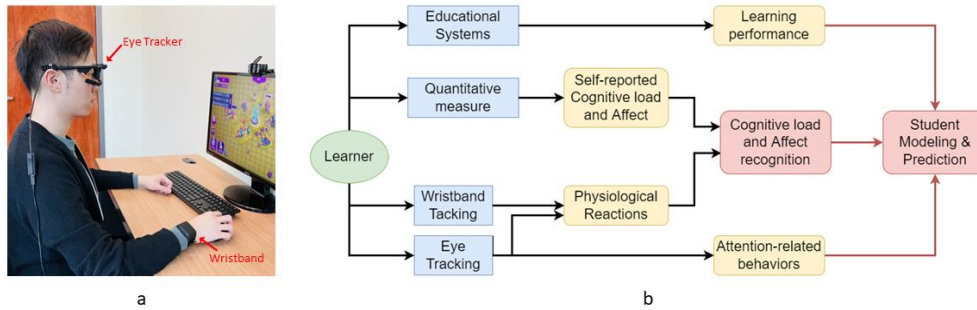


Figure 1. Study Overview: (a) shows the device set up and (b) shows the overall process

these loads when also including EDA data. Participant responses to affect and cognitive load scales will be used as a reference.

For the pupil data, we will use changes in PD instead of its absolute value as a feature to eliminate the influence of individual differences on model training. In a preprocessing step, we will calculate PD values as proportion change, by dividing pupil size by the grand mean PD size during the baseline period averaged across all attempts.

3.2 How does learner affect and cognitive load change alongside their learning when interacting with educational systems?

We will investigate the dynamics of cognitive load and affect across learning activities with a focus on what contributes to changes in cognitive load and affect. We are especially interested in understanding how the learning activity sequence and user interface design contribute to these latent states.

We would like to compare the continuous reactions of cognitive load and affect with information from learner behaviors and performance with educational technologies. Learner's interaction and learning performance information (i.e., score, task completion) will be extracted from the learning systems. Learner behavior data will come from two sources: the learning system's logs (i.e., keyboard input, buttons clicked, questions answered) and eye tracking (i.e., gaze point and trajectory).

We will investigate what kinds of information contributes to increasing cognitive load or changes in affect by comparing the eye information actions obtained by students at critical moments (the moment when cognitive load or affect change).

Ideally, we will compare the cognitive load and affect levels of students who had similar achievements. This analysis will be done across their learning session(s) to explore whether there is a specific pattern among them.

3.3 Can affect and cognitive load help in identifying a learner's knowledge state and trajectories?

We will use the models developed under section 3.1 and 3.2 to help predict a learner's knowledge and how it changes during learning. To answer this question, initial experiments will be conducted to predict a learner's knowledge level. The prediction of this information can be important for providing suitable learning materials and learning sequences. As an initial step, we will evaluate two approaches: one heuristic with collaborative filtering (CF) and the other machine learning (ML) using engineered features. Based on

the initial results, further experiments will be performed to identify changes in learner knowledge or mastery.

4. PROGRESS TO DATE & PLAN

So far, we have implemented our sensing platform and designed two case studies. The proposed case studies have been approved by our institutional Research Ethics Board (REB), and we have begun piloting the first.

4.1 Sensing System

In our literature search, we noticed there is no integrated sensing system to allow us to easily collect multiple types of learner physiological data. In this project, we choose multimodal analysis to support the recognition of cognitive load and affect. Thus, we need a sensing platform that enables us to collect and synchronize a learner's physiological features automatically. This generates databases of learners' physiological reactions that can be used to extract their cognitive load and affect characteristics. Our approach focuses on modeling with two dimensions of real-time physiological information: pupil dynamics collected by an eye tracker and cardiovascular and EDA data from non-invasive wearable sensors. As mentioned in Section 2, previous studies have shown the possibility of using pupil dynamics and electrodermal activity to investigate the continuous monitoring of cognitive load. These types of data have also been used to recognize state-based affect.

We developed our sensing platform that allows subjects to remain relatively comfortable (Figure 1-a). The platform consists of two non-invasive sensors and a software system that facilitates the synchronization of data from different channels. The first sensor is an open-source eye tracker (Pupil Core, Pupil Labs), which records dual eye movements at 200 Hz and includes gaze tracking and pupillometry. It is worn like normal glasses and consists of a scene camera that records what the user sees. The second sensor is a wireless wristband (E4, Empatica Inc.), which is worn as a wristwatch and connected via Bluetooth. The wristband includes photoplethysmography (PPG), electrodermal activity (EDA), skin temperature, and accelerometer sensors. The PPG sensor is used to measure blood volume pulse (BVP) which can then be used to calculate heart rate.

As these time series data come from different sources, they need to be synchronized to enable analysis. For this purpose, we have developed a system based on an open multi-model recording framework – Lab Streaming Layer (LSL). This system will be used to support real-time data streaming, synchronization, and recording with a laptop.

4.2 Case Studies

According to what we discussed in Section 3, our case studies will collect three types of learner data (Figure 1-b): (a) physiological

data from the sensing system, (b) self-report measures, and demographic information from questionnaires, and (c) learner interaction and performance from the learning system.

4.2.1 Experiment procedure

The experiment will take approximately 90 minutes. Sensor calibration will follow consent. Participants will interact with a specific e-learning system for approximately 60 minutes. Sensors will be used to collect learner data as they perform learning tasks in the system. Self-report instruments for measuring affect and cognitive load will be administered every 5-10 minutes. Demographics will be collected at the end.

4.2.2 Learning environments (LE)

The first study (S1) will be conducted when students are learning programming using the Mastery Grids e-learning system within our lab (Figure 2). This system visualizes learner progress and provides multiple types of interactive learning activities. An explicit, visually rich representation with social comparison helps students track their activities alongside those of their peers. Interactive learning content with feedback helps students practice and makes them aware of the knowledge they are expected to learn.

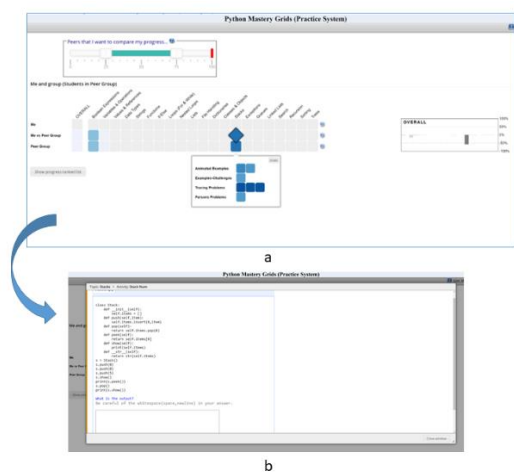


Figure 2 Online learning portal for python learning: (a) visualization of individual learning progress and social comparison (comparing the percentage of attempted units) and (b) a programming task with feedback.



Figure 3 In the strengthening of their role in the virtual game world, learners are motivated to solve reading tasks with appropriate difficulty levels: (a) game interface, (b) a passage, and (c) an English-language arts question.

The second study (S2) will be conducted when learners are using a web-based online learning game (Figure 3). It engages players to enhance literacy skills for English learning using a popular base-building game design. Players pretend to be in the realm of dreams and are tasked with defending their virtual home from invading “reveries” (in-game creatures). The game combines the strategy and engagement of the game factors with passage reading and comprehension tasks to create an interactive learning experience.

4.2.3 Participants

Different people will be recruited for each case study. The target participants in S1 will be 35 undergraduate students registered in an introductory computer science course that teaches how to program in python. The target participants in S2 (n = 35) will be students registered in an English as an additional language course.

4.2.4 Self-reported measures

In this project, self-report information will be used as verification of our sensor-based measurement. We follow the definition of CLT [33], dividing cognitive load into three types: intrinsic, extraneous, and germane. Based on prior work [19], we developed Likert-scale items to measure all three types of cognitive load.

Affect has been defined in many different ways, and no generally agreed upon definition has emerged [26, 27]. We follow Russell’s model [30] that describes affect in two dimensions – arousal and valence. We adopt an established scale – the international positive and negative affect schedule short-form (I-PANAS-SF) [36] to measure both the positive and negative affect of learners.

We will collect participants’ demographic information using a questionnaire at the end of the study session. In addition to basic information (e.g., age, gender) we will collect information about participants’ programming background for S1 and language-learning background (e.g., mother tongue, IELTS/TOEFL scores) for S2.

4.3 Future Work

In conclusion, we have implemented the sensing prototype and designed the case studies that will collect learner data for student modeling. We will conduct the case studies and perform the modeling work.

If modelling cognitive load and affect better explains student learning trajectories, we will incorporate these models so that they inform system adaptation, thus optimizing student learning experience and performance.

5. ACKNOWLEDGEMENTS

This work is supported in part by funding from the Social Sciences and Humanities Research Council of Canada and the Natural Sciences and Engineering Research Council of Canada (NSERC), [RGPIN-2018-03834].

6. REFERENCES

- [1] Antonenko, P., Paas, F., Grabner, R. and van Gog, T. 2010. Using Electroencephalography to Measure Cognitive Load. *Educational Psychology Review*. 22, 4 (Dec. 2010), 425–438. DOI:https://doi.org/10.1007/s10648-010-9130-y.
- [2] Baddeley, A., Logie, R., Bressi, S., Sala, S.D. and Spinnler, H. 1986. Dementia and Working Memory. *The Quarterly Journal of Experimental Psychology Section A*. 38, 4 (Nov. 1986), 603–618. DOI:https://doi.org/10.1080/14640748608401616.
- [3] Beatty, J. 1982. Task-evoked pupillary responses, processing load, and the structure of processing resources. *Psychological*

- Bulletin*. 91, 2 (1982), 276–292. DOI:https://doi.org/10.1037/0033-2909.91.2.276.
- [4] Bless, H. and Fiedler, K. 2006. Mood and the regulation of information processing and behavior. *Affect in social thinking and behavior*. Psychology Press. 65–84.
- [5] Booth, J.L., Lange, K.E., Koedinger, K.R. and Newton, K.J. 2013. Using example problems to improve student learning in algebra: Differentiating between correct and incorrect examples. *Learning and Instruction*. 25, (Jun. 2013), 24–34. DOI:https://doi.org/10.1016/j.learninstruc.2012.11.002.
- [6] Bounajim, D., Rachmatullah, A., Hinckle, M., Mott, B., Lester, J., Smith, A., Emerson, A., Morshed Fahid, F., Tian, X., Wiggins, J.B., Elizabeth Boyer, K. and Wiebe, E. 2021. Applying Cognitive Load Theory to Examine STEM Undergraduate Students’ Experiences in An Adaptive Learning Environment: A Mixed-Methods Study. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*. 65, 1 (Sep. 2021), 556–560. DOI:https://doi.org/10.1177/1071181321651249.
- [7] Brand, S., Reimer, T. and Opwis, K. 2007. How do we learn in a negative mood? Effects of a negative mood on transfer and learning. *Learning and Instruction*. 17, 1 (Feb. 2007), 1–16. DOI:https://doi.org/10.1016/j.learninstruc.2006.11.002.
- [8] Brunken, R., Plass, J.L. and Leutner, D. 2003. Direct Measurement of Cognitive Load in Multimedia Learning. *Educational Psychologist*. 38, 1 (Mar. 2003), 53–61. DOI:https://doi.org/10.1207/S15326985EP3801_7.
- [9] Brünken, R., Plass, J.L. and Moreno, R. eds. 2010. Current issues and open questions in cognitive load research. *Cognitive load theory*. Cambridge University Press. 253–272.
- [10] Craig, S., Graesser, A., Sullins, J. and Gholson, B. 2004. Affect and learning: An exploratory look into the role of affect in learning with AutoTutor. *Journal of Educational Media*. 29, 3 (Oct. 2004), 241–250. DOI:https://doi.org/10.1080/1358165042000283101.
- [11] Fraser, K., Huffman, J., Ma, I., Sobczak, M., McIlwrick, J., Wright, B. and McLaughlin, K. 2014. The emotional and cognitive impact of unexpected simulated patient death: a randomized controlled trial. *Chest*. 145, 5 (May 2014), 958–963. DOI:https://doi.org/10.1378/chest.13-0987.
- [12] Fraser, K.L., Ayres, P. and Sweller, J. 2015. Cognitive Load Theory for the Design of Medical Simulations. *Simulation in Healthcare*. 10, 5 (Oct. 2015), 295–307. DOI:https://doi.org/10.1097/SIH.0000000000000097.
- [13] Hart, S.G. and Staveland, L.E. 1988. Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research. *Advances in Psychology*. P.A. Hancock and N. Meshkati, eds. North-Holland. 139–183.
- [14] Kalyuga, S. and Singh, A.-M. 2016. Rethinking the Boundaries of Cognitive Load Theory in Complex Learning. *Educational Psychology Review*. 28, 4 (Dec. 2016), 831–852. DOI:https://doi.org/10.1007/s10648-015-9352-0.
- [15] Klingner, J., Tversky, B. and Hanrahan, P. 2011. Effects of visual and verbal presentation on cognitive load in vigilance, memory, and arithmetic tasks. *Psychophysiology*. 48, 3 (2011), 323–332. DOI:https://doi.org/10.1111/j.1469-8986.2010.01069.x.
- [16] Knörzer, L., Brünken, R. and Park, B. 2016. Facilitators or suppressors: Effects of experimentally induced emotions on multimedia learning. *Learning and Instruction*. 44, (Aug. 2016), 97–107. DOI:https://doi.org/10.1016/j.learninstruc.2016.04.002.
- [17] Koedinger, K.R. and Alevan, V. 2007. Exploring the Assistance Dilemma in Experiments with Cognitive Tutors. *Educational Psychology Review*. 19, 3 (Sep. 2007), 239–264. DOI:https://doi.org/10.1007/s10648-007-9049-0.
- [18] LeDoux, J.E. and Brown, R. 2017. A higher-order theory of emotional consciousness. *Proceedings of the National Academy of Sciences*. 114, 10 (Mar. 2017), E2016–E2025. DOI:https://doi.org/10.1073/pnas.1619316114.
- [19] Leppink, J., Paas, F., van Gog, T., van der Vleuten, C.P.M. and van Merriënboer, J.J.G. 2014. Effects of pairs of problems and examples on task performance and different types of cognitive load. *Learning and Instruction*. 30, (Apr. 2014), 32–42. DOI:https://doi.org/10.1016/j.learninstruc.2013.12.001.
- [20] Mayer, R.E. and Moreno, R. 2003. Nine Ways to Reduce Cognitive Load in Multimedia Learning. *Educational Psychologist*. 38, 1 (Mar. 2003), 43–52. DOI:https://doi.org/10.1207/S15326985EP3801_6.
- [21] Moreno, R. 2010. Cognitive load theory: more food for thought. *Instructional Science*. 38, 2 (Mar. 2010), 135–141. DOI:https://doi.org/10.1007/s11251-009-9122-9.
- [22] Paas, F. and Ayres, P. 2014. Cognitive Load Theory: A Broader View on the Role of Memory in Learning and Education. *Educational Psychology Review*. 26, 2 (Jun. 2014), 191–195. DOI:https://doi.org/10.1007/s10648-014-9263-5.
- [23] Paas, F., Camp, G. and Rikers, R. 2001. Instructional compensation for age-related cognitive declines: Effects of goal specificity in maze learning. *Journal of Educational Psychology*. 93, 1 (2001), 181–186. DOI:https://doi.org/10.1037/0022-0663.93.1.181.
- [24] Paas, F., Tuovinen, J.E., Tabbers, H. and Van Gerven, P.W.M. 2003. Cognitive Load Measurement as a Means to Advance Cognitive Load Theory. *Educational Psychologist*. 38, 1 (Mar. 2003), 63–71. DOI:https://doi.org/10.1207/S15326985EP3801_8.
- [25] Paas, F.G.W.C. and Van Merriënboer, J.J.G. 1994. Instructional control of cognitive load in the training of complex cognitive tasks. *Educational Psychology Review*. 6, 4 (Dec. 1994), 351–371. DOI:https://doi.org/10.1007/BF02213420.
- [26] Pessoa, L. 2008. On the relationship between emotion and cognition. *Nature Reviews Neuroscience*. 9, 2 (Feb. 2008), 148–158. DOI:https://doi.org/10.1038/nrn2317.
- [27] Plass, J.L. and Kaplan, U. 2016. Emotional Design in Digital Media for Learning. *Emotions, Technology, Design, and Learning*. S.Y. Tettegah and M. Gartmeier, eds. Academic Press. 131–161.
- [28] Raptis, G.E., Katsini, C., Belk, M., Fidas, C., Samaras, G. and Avouris, N. 2017. Using Eye Gaze Data and Visual Activities to Infer Human Cognitive Styles: Method and Feasibility Studies. *Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization* (New York, NY, USA, Jul. 2017), 164–173.
- [29] Rogers, K.B. and Robinson, D.T. 2014. Measuring Affect and Emotions. *Handbook of the Sociology of Emotions: Volume II*. J.E. Stets and J.H. Turner, eds. Springer Netherlands. 283–303.
- [30] Russell, J.A. 2003. Core affect and the psychological construction of emotion. *Psychological Review*. 110, 1 (2003), 145–172. DOI:https://doi.org/10.1037/0033-295X.110.1.145.
- [31] Salomon, G. 1984. Television is “easy” and print is “tough”: The differential investment of mental effort in learning as a function of perceptions and attributions. *Journal of Educational Psychology*. 76, 4 (1984), 647–658. DOI:https://doi.org/10.1037/0022-0663.76.4.647.
- [32] Sutton, E., Youssef, Y., Meenaghan, N., Godinez, C., Xiao, Y., Lee, T., Dexter, D. and Park, A. 2010. Gaze disruptions experienced by the laparoscopic operating surgeon. *Surgical*

- Endoscopy*. 24, 6 (Jun. 2010), 1240–1244. DOI:<https://doi.org/10.1007/s00464-009-0753-3>.
- [33] Sweller, J. 1988. Cognitive load during problem solving: Effects on learning. *Cognitive Science*. 12, 2 (Apr. 1988), 257–285. DOI:[https://doi.org/10.1016/0364-0213\(88\)90023-7](https://doi.org/10.1016/0364-0213(88)90023-7).
- [34] Sweller, J. 2010. Element Interactivity and Intrinsic, Extraneous, and Germane Cognitive Load. *Educational Psychology Review*. 22, 2 (Jun. 2010), 123–138. DOI:<https://doi.org/10.1007/s10648-010-9128-5>.
- [35] Sweller, J., van Merriënboer, J.J.G. and Paas, F.G.W.C. 1998. Cognitive Architecture and Instructional Design. *Educational Psychology Review*. 10, 3 (Sep. 1998), 251–296. DOI:<https://doi.org/10.1023/A:1022193728205>.
- [36] Thompson, E.R. 2007. Development and Validation of an Internationally Reliable Short-Form of the Positive and Negative Affect Schedule (PANAS). *Journal of Cross-Cultural Psychology*. 38, 2 (Mar. 2007), 227–242. DOI:<https://doi.org/10.1177/0022022106297301>.
- [37] Tyng, C.M., Amin, H.U., Saad, M.N.M. and Malik, A.S. 2017. The Influences of Emotion on Learning and Memory. *Frontiers in Psychology*. 8, (2017), 1454.
- [38] Van Gerven, P.W.M., Paas, F.G.W.C., Van Merriënboer, J.J.G. and Schmidt, H.G. 2002. Cognitive load theory and aging: effects of worked examples on training efficiency. *Learning and Instruction*. 12, 1 (Feb. 2002), 87–105. DOI:[https://doi.org/10.1016/S0959-4752\(01\)00017-2](https://doi.org/10.1016/S0959-4752(01)00017-2).
- [39] Zimmermann, P. 2008. Beyond usability: measuring aspects of user experience. (2008).

Using AI, ML and Sentiment Analysis to Increase Diversity and Equity in Technology Training and Careers

Jonathan Precey Young, Sue Black, Alexandra Cristea, Cristina Todor, Ryan Hodgson

Durham University

jonathan.p.young@durham.ac.uk, sue.black@durham.ac.uk, alexandra.i.cristea@durham.ac.uk,
cristina.r.todor@durham.ac.uk, ryan.t.hodgson@durham.ac.uk

ABSTRACT

We present a new dataset which captures the sentiment and thoughts of applicants to an industrial IT training course in relation to *technology careers* and in relation to *technology usage*. The dataset is live and capturing over 2,000 new applicants' data each week, from 12 countries, and contains a wide range of diversity information such as *gender*, *ethnicity*, and *sexuality*. The applicants are almost all a similar age, career status and education status, thereby limiting the demographic variables in the dataset.

This dataset has been created in collaboration with a technology consulting firm for use in this PhD research into increasing diversity and equity in technology careers and training. Research into how people from different demographics think about technology careers and technology usage and these topics side-by-side will be facilitated by this dataset. It is intended that relevant findings from the research will be applied at the technology consulting firm to increase equity and diversity.

The data are anonymised during the upload to the dataset and the data provenance is achieved through each upload being controlled, audited and fully repeatable.

It is the intention that a version of this dataset will be made public.

Keywords

Diversity, technology careers, public dataset.

1. INTRODUCTION

The substantial research into the low uptake of technology careers amongst underrepresented groups agrees that the representation of these groups in technology careers remains too low and has changed little despite considerable effort to try to affect change[6, 20, 25]. Use of technology such as internet and smartphones is now widespread across all demographics in society[27]. Research has shown that the diversity of users of technology has evolved, and, in the USA and Europe, we are in the third phase of the Digital Divide[27].

We have observed from working with the graduates who join the technology consultancy's training programme that they show confidence with and ownership of their technology, whether

J. Young, S. Black, A. Cristea, R. Hodgson, and C. Todor. Using AI, ML and sentiment analysis to increase diversity and equity in technology training and careers. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 805–810, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852992>

mobile phone or Personal Computer (PC). This is often at odds with the lack of confidence with which they speak about the prospect and challenge of being active in creating and building technology.

This research will use ML/AI, sentiment analysis and Natural Language Processing (NLP) to understand patterns in the sentiment of different groups in society, including underrepresented demographic groups, to technology use and to technology careers and investigate whether this contributes to the continued absence of diversity in people in technology careers. Our focus will be an application of these ML/AI, sentiment analysis and NLP techniques and tools to the dataset gathered from the recruitment process of a global technology consultancy and enhanced with free text responses. The dataset will be extended by ~100,000 rows each year through the normal volume of applications for the training programme. Alongside analysing data patterns in sentiment across different groups in relation to both technology usage and technology careers, we will also examine the evolution of sentiment amongst graduates who are undertaking the consultancy's training for a career in technology.

This paper describes the **Graduate Technology Applicant Dataset (GTAD)** which contains the applicants' diversity data as well their free text responses to questions related to their thoughts on *technology usage* and *technology careers*. The diversity data includes, but is not limited to, *gender*, *ethnicity*, *age*, *school type*, *sexual orientation*, and whether the applicant was the *first in their family to go to university*. The questions have been designed to allow the respondent to write freely about their thoughts at the time of application. Growing at around 2,000 records per week and covering applicants from 12 countries, GTAD also receives around 2,000 responses to the free text questions each week.

This paper describes how the data provenance is maintained and gives some preliminary research findings using statistical methods, sentiment analysis and text analysis tools. The paper outlines a version of GTAD which could be made publicly available in the future.

RQ: Are there differences in perception of technology careers and technology usage for different demographic groups (e.g., women compared to men, 'pale and male' respondents compared to everyone else) which can be modelled?

To be able to address this question, the following objectives (Ox) have been defined:

O1: Create GTAD, collecting long text perceptions on technology careers and usage. As the related research shows, there is an absence of such datasets in the literature. In this paper, we mostly focus on objective O1 and how we create this dataset.

O2: Design experiments related to the demographics targeted, using the dataset created in O1. Preliminary results of some initial experiments are also outlined in this paper.

2. RELATED RESEARCH and WORK

The lack of gender diversity in technology careers has been an active area of research for many years[17, 21, 24], wherein it is agreed that the composition of the technology workforce does not reflect the diversity in society[6, 20]. The findings of Spertus [17] in 1991 in relation to women in IT, such as lack of role models, male-dominated work culture, and lack of workplace flexibility, were repeated in many of the 213 entries in Trauth’s Encyclopedia of Gender and IT[22] in 2006 and are found in Vainionpää’s work [23] in 2021, with much of the literature reporting findings from interviews and surveys of modest sized groups of people from specific, common demographics. Our research adopts a new approach of researching technology career perceptions using analysis of a large, tailored dataset of long text responses from multiple demographic groups.

Van Dijk notes that research into the digital divide doesn’t agree on a model but does agree that the digital divide is changing and his research agrees with Marler and the Office for National Statistics that the uptake and use of technology has diversified considerably and is more widespread [10, 19, 27]. This suggests that perceptions of technology usage have evolved, unlike perceptions of technology careers, which is why we are collecting perceptions on both topics for side-by-side research.

In terms of methodology useful for researching demographic perception diversity in technology careers, there is substantial research into NLP and sentiment analysis to analyse large bodies of data as described in the comprehensive survey by Pang and Lee in 2008[11] and in the scholars’ introduction to the subject by Liu in 2010[8]. Recent relevant research by Yu[4] has included sentiment analysis of attendees of a technology training course. Another recent research used statistics and tools such as Linguistic Inquiry and Word Count (LIWC)[14] to analyse emotional, structural, and cognitive components of long text datasets to comparing gender differences in the way people worry about COVID-19[26]. This research contains results of experiments on data in the Real-World Worry Dataset (RWWD) gathered from 2,500 respondents who wrote their thoughts on COVID-19. They demonstrated differences in the way men and women thought about COVID-19 at that time. The paper concluded that longer statements were more relevant and accurate than short tweets and called for more publicly available high-quality datasets rather than relying on social media datasets. Therefore, our research has created a new, long text dataset, specifically designed for capturing diverse perceptions in relation technology *careers* and *usage* and we propose to apply these techniques and tools to that dataset to look for patterns and themes.

3. CREATION OF THE DATASET

3.1 The Need for GTAD

As shown in section 2, a new dataset is necessary, with as many responses as possible, from a wide diversity of applicants commenting on their thoughts about technology careers and technology usage.

Creating a dataset using similar ‘long text’ responses as used in the RWWD but containing questions on *technology usage* and *technology careers* could replicate some of this methodology to examine whether there are statistically significant differences in

responses – women compared to men, white respondents in relation to people of colour, *pale and male* respondents compared to everyone else. A dataset with as many diversity data points as possible could lead to multiple options for comparison in the future and can be made publicly available.

3.2 GTAD Methodology

Digital Divide research is the main area focus of our literature review in relation to IT usage. More than half the research into the Digital Divide comes from USA and most of the research originates in English-speaking developed and (relatively) rich countries[27]. This correlates with the origination of the majority of the work in relation to diversity in tech careers and, for these reasons the focus of our research is on the UK and USA[15]. There is not, to the best of our knowledge, a publicly available database containing data such as the data in GTAD.

GTAD’s main purpose is to capture free text responses on how the applicants feel about technology usage and technology careers. The data is collected from the technology consultancy’s Applicant Tracking System (ATS) and contains diversity related data such as *gender, ethnicity, sexuality, first to go to university, disability*. GTAD also contains data related to the *application process* and the *outcomes* of that process such as whether the candidate was rejected or withdrew and at what stage of the process.

Data collection started on May 22nd, 2018, from USA, Canada, UK, Germany, Poland, China, Hong Kong, Singapore, Australia, Ireland, South Africa, Luxemburg, Netherlands. Although the focus of this PhD research is UK and USA, the data for all other geographies will also continue to be captured to make GTAD as rich as possible for future research.

The free text questions were added to the system on August 1st, 2021, replicating the long word format from the RWWD paper[26], and replacing “*the Corona situation*” with our research topics of *technology careers* and *technology usage* as shown below:

Question 1 (**careers**):
“Write in a few sentences how you feel about **starting a career in technology/IT** at this very moment. This text should express your feelings at this moment.”

Question 2 (**usage**):
“Write in a few sentences how you feel about **using technology/IT** (e.g., smartphone/PC/Wi-Fi) at this very moment. This text should express your feelings at this moment.”

On the same ATS screen, alongside the questions, it is made clear that the data is to be used for research into diversity in technology careers, will be anonymised and has no bearing on the candidate’s application.

GTAD is populated with secondary data and is anonymised from the original dataset so has limited relevance in terms of data privacy. Nonetheless, approval was sought and gained from the Executive of the consultancy for the data to be used for research. Approval from the Ethics review board of Durham University for the use of GTAD for research was sought on February 15th, 2022.

It is the intention that a subset of this dataset, less any commercially sensitive information, will be made public for other researchers to use. An early instance of this proposed public dataset has been used for MSc level experiments at Durham University.

3.3 GTAD Population

For the GTAD dataset defined below, at least 91% of UK applications are from university graduates. All are at a similar stage in their career and considering a career and training in technology, with 91.1% of the current population who provided their age are aged 18-30. This limited variation in the dataset population could be classified as bias. At the same time, the similarities could ensure that the effect of different diversity characteristics are more pronounced. Combined with the various diversity data fields, GTAD could be a useful research asset when made public.

For the period up to January 1st, 2022, GTAD comprised:

- 565,937 applications from candidates, over 44,000 of which were received since the questions were added to the application form.
- 25,243 long text questions answers, spread 12,242 and 13,001 across *usage* and *careers* respectively.
- A corpus of 1,243,578 words.

The growth of GTAD has been constant throughout the period with an average of 2,000 applications per week, of which around 1,000 applicants are currently answering the free text questions, giving 2,000 answers and 100,000 words each week.

This growth rate trajectory to the end of this research (January 2027) suggests that the GTAD population could be substantial with over a million applications, over half a million free text questions answers and a corpus of over 27 million words.

3.4 Data Provenance

Via a series of initial data uploads from September 16th, 2021, to December 22nd, 2021, the upload process was tested and refined incrementally. The process is now controlled and repeatable and the scripts and code used to perform the uploads, as well as data manipulation and enrichment scripts, are stored and controlled using source code control tools and processes as prescribed by the technology consultancy's ISO-27001 compliant[3] IT policies and practices.

It is important that any information related to the status of the application is a **final** status i.e., at the end of the recruitment process as this relates to outcomes of the process. For example, Did the candidate withdraw? Were they successful in their application? If they were rejected, at what stage was this? The recruitment process lasts a maximum of 4 weeks and so, only data older than 5 weeks was included an initial GTAD upload of February 24th, 2022, covering the period, May 1st, 2018, to January 1st, 2022. Incremental uploads and will be repeated monthly, five weeks in arrears, following the subsequent upload of March 15th, 2022.

3.5 Data Enrichment and Optimisation

Some applicants to the training programme apply more than once and so the ATS contains multiple applications from some applicants. Where this is the case, the applicant may answer the questions in different ways for different applications. They may also elect to complete different elements of the diversity data in different applications. So, for example, they might enter no data in their first application. They then provide gender and ethnicity in the second application. In their third and final application they may choose to provide their sexuality and their education information. The uploads of the dataset include logic to ensure that the superset of all diversity related data for that applicant is

captured for each application. By capturing every individual application for every applicant, GTAD contains all responses to the technology questions at different times.

3.5.1 Future Enrichment

3.5.1.1 Predicting Gender and Ethnicity

The primary database (ATS) contains the names of applicants. It is intended to use the first name to enrich the data during the process of the upload and create derived attributes of ethnicity and gender, initial research into which is described later in this paper.

3.5.1.2 Anonymisation

Although the data is anonymised *en route* to GTAD by the removal of the identifying fields which are used in the ATS such as surname and date of birth, identifying data does appear in the dataset occasionally. One of the main ways in which this happens is when a contributor uses their name in their free text response. It is intended to use automated tools to anonymise the data during the upload to GTAD.

4. RESEARCH

4.1 Initial Research

Initial experiments were conducted on the data, some of which are summarised here.

4.1.1 Recruitment Process Outcomes Analysis

A comparison of the 10 of the most relevant recruitment process outcomes (from a total of 48 outcomes) such as 'Rejected After Assessment Day', 'Rejected After Video/Phone Interview' and 'Invited to Video/Phone Interview', in the UK revealed statistically significant patterns in the outcomes for different diversity groups. The recruitment process is reviewed and certified annually as fair by a 'Big Four' consultancy firm and so these differences in outcomes for specific demographic groups was an unexpected outcome.

4.1.2 Gender Prediction

A dataset was created according to the method described here with the added step that the applications' first name was retained. For the non-NULL values, the `genderizeR`[28] and `Nltk`[9] libraries were used to predict gender and the former was found to be most effective. Globally, `genderizeR` was 86% accurate and for the USA and UK, 89% and 88% respectively (see Table 1. Gender Prediction Performance). Figures 1,2 show the respective confusion matrices for these countries.

Table 1. Gender Prediction Performance

Methodology	Accuracy	Precision	Recall	F_1 Score
GenderizeR (USA)	89%	0.88	0.94	0.91
GenderizeR (UK)	88%	0.86	0.93	0.90
GenderizeR (Global)	86%	0.85	0.79	0.88
Nltk (Global)	67%	0.75	0.69	0.70

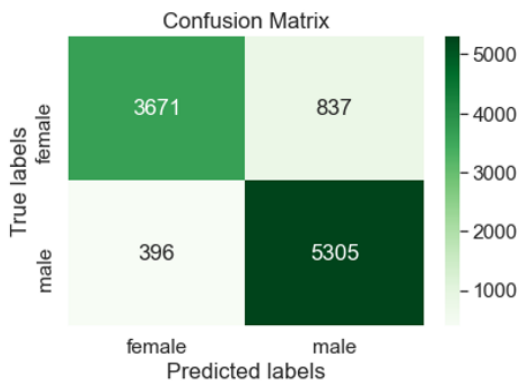


Figure 1. GenderizeR Performance UK

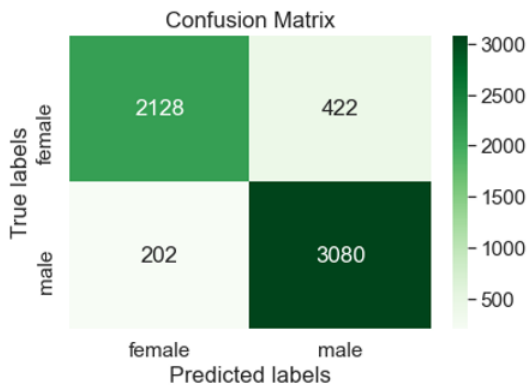


Figure 2. GenderizeR Performance USA

4.1.3 Sentiment Analysis

Sentiment analysis was conducted using the Flair Python library[7, 18], returning a sentiment category (positive, negative, neutral) and a score. This analysis revealed mostly positive sentiment (*career*: 96.63% and *usage*: 89.69% positive). There was no neutral sentiment and entries from ‘not disclosed’ applications made up 30% of *career* and 68.4% of *usage* responses classed as negative.

4.1.4 K-means Clustering

K-means clustering was performed using the scikit-learn library[13]. The optimal number of clusters was found by the elbow method to be 3. Features resulted in the highest score of 0.878 were *location*, *ethnicity*, and *stream* (i.e., technical training course) chosen by the applicant.

4.1.5 LIWC Analysis

LIWC 2015[14] analysis was performed on the applicants’ long text responses, comparing the results for different groups to the results from the most dominant group in technology careers, white males. Non-parametric Kruskal-Wallis was used to determine differences which were statistically significant, and Cohen’s *d* was used to determine the magnitude of the differences.

There were differences between other groups and white males for *usage*, but these were not as pronounced as the differences in the responses related to *careers*. In the latter, white men focus more on professional aspects, by being more analytical and touching on subjects such as *time*, *money*, *numbers*, *power*, and *work*. Whereas other groups tend to share more personal thoughts speaking about *feelings*, *drives*, *family* and showing an emotional tone. White men also use more words per sentence

when speaking about *careers* and are more authentic, speaking about *certainty* whereas the other groups seem to share personal experience, speaking about *affect*, *affiliation*, and *rewards*.

4.1.6 Top2Vec Analysis

Topic modelling of questionnaire responses was performed, facilitated through Top2Vec[2]. This approach seeks to identify dense clusters of document vectors using hierarchical clustering techniques[5]. Based on an analysis of all responses to question 1 of the questionnaire, 16 distinct topics were identified from the topic modelling approach, shown with the first 5 words in each in Figure 3. These require a manual qualitative analysis when defining a topic label, with the topics demonstrating responses relevant to *fast-paced work environments*; *educational background*; and *career growth* when manually analysed and labelled (topic ids 0, 1, 6 respectively). This demonstrates the initial feasibility of the approach, which may be subsequently investigated further with the aims of identifying more coherent topics from the data.

		MOST PREVALENT WORDS - decreasing frequency with higher number					
		0	1	2	3	4	5
TOPIC ID	0	evolving	constantly	fast	change	excites	exciting
	1	university	computer	science	coding	degree	programming
	2	forward	looking	apply	real	problems	look
	3	if	not	but	much	know	just
	4	projects	coding	apply	eager	programming	professional
	5	pandemic	we	our	covid	important	becoming
	6	growth	industries	their	companies	innovation	society
	7	wanted	feeling	long	company	get	ve
	8	solving	problem	enjoy	skills	challenging	challenge
	9	important	our	becoming	society	therefore	we
	10	professional	gain	am	passionate	looking	fdm
	11	challenges	new	forward	ready	challenge	eager
	12	cloud	computing	technical	related	interested	data
	13	us	keep	solutions	improve	lives	environment
	14	ready	challenge	put	new	pursue	challenges
	15	forward	looking	moment	feeling	ready	start
16	confident	put	strong	start	fdm	technical	

Figure 3. Top2Vec Topic Clustering

4.2 Future Research

The immediate focus of the research is on the enhancement and enrichment of GTAD as the baseline for the analysis and experimentation. Further analysis of the enhanced dataset plus qualitative analysis, through interview and workshops with the consultancy’s recruiters, will be used to research the clustering of outcomes for different groups in the technology consultancy’s recruitment process. Derived ethnicity attribute via ethnicolr[16] will be further researched as well as anonymization of data using tools such as Anonymate[1].

Sentiment analysis and textual analysis tools will then be tested against the data for patterns in the text across diverse groups and the LIWC analysis could be tuned and improved using the latest version of the tool. Improvements in the Top2Vec topic modelling performance could be researched and analysis of the frequencies of demographic groups within specific topics may help in the identification of attitudes towards questionnaire responses and may aid in future works.

Since the researcher’s relationship with both the dataset and the recruitment processes is not passive, we intend to experiment with causal inference[12] and interventions in the process and data.

5. CONCLUSION AND OUTLOOK

There was a consistent rate of response to the questions on *technology usage* and *technology careers* from applicants to the technology consultancy’s training course, both in terms of numbers of applicants responding and length of responses. This

suggests that GTAD will continue to grow, both in size and richness, and could be used for research including research into the PhD hypotheses of the author.

This work is at an early stage and any feedback or ideas on the paper would be highly welcomed. Particularly of interest would be comments on what research and analysis could be performed using the data, and thoughts on the wider distribution of the non-commercially sensitive dataset.

6. ACKNOWLEDGMENTS

We acknowledge the contribution of Stuart Brown from the IT consultancy's technology team for his advice on database design and his implementation of the data extraction code for GTAD.

7. REFERENCES

- [1] ADAMS, A., AILI, E., AIOANEI, D., JONSSON, R., MICKELSSON, L., MIKMEKOVA, D., ROBERTS, F., VALENCIA, J.F., and WECHSLER, R., 2019. AnonyMate: A toolkit for anonymizing unstructured chat data. In *Proceedings of the Workshop on NLP and Pseudonymisation*, 1-7.
- [2] ANGELOV, D., 2020. Top2vec: Distributed representations of topics. *arXiv preprint arXiv:2008.09470*. DOI=<http://dx.doi.org/https://doi.org/10.48550/arXiv.2008.09470>.
- [3] BRENNER, J., 2007. ISO 27001 risk management and compliance. *Risk management* 54, 1, 24-29.
- [4] BUCK, G.A., CLARK, V.L.P., LESLIE-PELECKY, D., LU, Y., and CERDA-LIZARRAGA, P., 2008. Examining the cognitive processes used by adolescent girls and women scientists in identifying science role models: A feminist approach. *Science Education* 92, 4, 688-707.
- [5] CAMPELLO, R.J., MOULAVI, D., and SANDER, J., 2013. Density-based clustering based on hierarchical density estimates. In *Pacific-Asia conference on knowledge discovery and data mining* Springer, 160-172. DOI=http://dx.doi.org/https://doi.org/10.1007/978-3-642-37456-2_14.
- [6] HOLTZBLATT, K., BALAKRISHNAN, A., EFFNER, T., RHODES, E., and TUAN, T., 2016. Beyond the pipeline: addressing diversity in high tech. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, 1063-1068. DOI=<http://dx.doi.org/https://doi.org/10.1145/2851581.2886422>.
- [7] KULA, S., CHORAŚ, M., KOZIK, R., KSIENIEWICZ, P., and WOŹNIAK, M., 2020. Sentiment analysis for fake news detection by means of neural networks. In *International Conference on Computational Science* Springer, 653-666. DOI=http://dx.doi.org/https://doi.org/10.1007/978-3-030-50423-6_49.
- [8] LIU, B., 2010. *Sentiment Analysis and Subjectivity*.
- [9] MAMGAIN, S., BALABANTARAY, R.C., and DAS, A.K., 2019. Author profiling: Prediction of gender and language variety from document. In *2019 International Conference on Information Technology (ICIT)* IEEE, 473-477. DOI=<http://dx.doi.org/https://doi.org/10.1109/ICIT48102.2019.00089>.
- [10] MARLER, W., 2018. Mobile phones and inequality: Findings, trends, and future directions. *New media & society* 20, 9, 3498-3520. DOI=<http://dx.doi.org/https://doi.org/10.1177/1461444818765154>.
- [11] PANG, B. and LEE, L., 2008. Opinion mining and sentiment analysis. *Foundations and Trends® in information retrieval* 2, 1-2, 1-135. DOI=<http://dx.doi.org/http://dx.doi.org/10.1561/1500000011>.
- [12] PEARL, J., 2010. Causal inference. *Causality: objectives and assessment*, 39-58.
- [13] PEDREGOSA, F., VAROQUAUX, G., GRAMFORT, A., MICHEL, V., THIRION, B., GRISEL, O., BLONDEL, M., PRETTENHOFER, P., WEISS, R., and DUBOURG, V., 2011. Scikit-learn: Machine learning in Python. *the Journal of machine Learning research* 12, 2825-2830.
- [14] PENNEBAKER, J.W., BOYD, R.L., JORDAN, K., and BLACKBURN, K., 2015. *The development and psychometric properties of LIWC2015*.
- [15] REDDOCK, R., MCFEE, D., and RADIX, C.A., 2008. Encyclopedia of Gender and Information Technology, edited by Eileen M. Trauth. 2 vols. Hershey, PA: The Idea Group, 2006. 1451 pp. \$525.00 cloth. ISBN: 1-59140-815-6. *The Information Society* 24, 2 (2008/03/04), 128-131. DOI=<http://dx.doi.org/10.1080/01972240701884003>.
- [16] SOOD, G. and LAOHAPRAPANON, S., 2018. Predicting race and ethnicity from the sequence of characters in a name. *arXiv preprint arXiv:1805.02109*. DOI=<http://dx.doi.org/https://doi.org/10.48550/arXiv.1805.02109>.
- [17] SPERTUS, E., 1991. Why are there so few female computer scientists?
- [18] SRIRAM, A., LI, Y., and HADAEGH, A., 2021. Mining Social Media to Understand User Opinions on IoT Security and Privacy. In *2021 IEEE International Conference on Smart Computing (SMARTCOMP)* IEEE, 252-257. DOI=http://dx.doi.org/DOI=https://doi.org/10.1007/978-3-030-50423-6_49.
- [19] Office for National Statistics, 2020. Internet Access - households and Individuals. <https://www.ons.gov.uk/peoplepopulationandcommunity/householdcharacteristics/homeinternetandsocialmediausage/datasets/internetaccesshouseholdsandindividualsreference tables>
- [20] Office for National Statistics., 2020. Population Characteristic Research Tables. <https://www.ons.gov.uk/peoplepopulationandcommunity/populationandmigration/populationestimates/datasets/populationcharacteristicsresearchtables>
- [21] TRAUTH, E., 2017. A research agenda for social inclusion in information systems. *ACM SIGMIS Database: the DATABASE for Advances in Information Systems* 48, 2, 9-20. DOI=<http://dx.doi.org/https://doi.org/10.1145/3084179.3084182>
- [22] TRAUTH, E.M., 2006. *Encyclopedia of gender and information technology*. IGI Global.
- [23] VAINIONPÄÄ, F., 2021. GIRLS' CHOICES OF IT CAREERS A NEXUS ANALYTIC INQUIRY. In *FACULTY OF INFORMATION TECHNOLOGY AND ELECTRICAL ENGINEERING UNIVERSITY OF OULU*, 148.
- [24] VAINIONPÄÄ, F., KINNULA, M., IIVARI, N., and MOLIN-JUUSTILA, T., 2019. Career Choice and Gendered Perceptions of IT—A Nexus Analytic Inquiry. In *International Conference on Information Systems*

- Development* Springer, 37-56. DOI=
http://dx.doi.org/10.1007/978-3-030-49644-9_3.
- [25] VAINIONPÄÄ, F., KINNULA, M., IIVARI, N., and MOLIN-JUUSTILA, T., 2019. GIRLS'CHOICE-WHY WON'T THEY PICK IT?
- [26] VAN DER VEGT, I. and KLEINBERG, B., 2020. Women worry about family, men about the economy: Gender differences in emotional responses to COVID-19. In *International Conference on Social Informatics* Springer, 397-409. DOI=
http://dx.doi.org/https://doi.org/10.1007/978-3-030-60975-7_29.
- [27] VAN DIJK, J., 2020. *The digital divide*. John Wiley & Sons.
- [28] WAIS, K., 2016. Gender prediction methods based on first names with genderizeR. *R J.* 8, 1, 17.

Effect of Q-matrix Misspecification on Variational Autoencoders (VAE) for Multidimensional Item Response Theory (MIRT) Models Estimation

Mahbubul Hasan, Lih Y Deng, John Sabatini, Dale Bowman, Ching-chi Yang, John Michael Hollander
University of Memphis, Memphis, TN, 38152, USA
[mhasan1, lihdeng, jpsbtini, ddbowman, cyang3, jmhlndr]@memphis.edu

ABSTRACT

Deep generative models with a specific variational autoencoding structure are capable of estimating parameters for the multidimensional logistic 2-parameter (ML2P) model in item response theory. In this work, we incorporated Q-matrix and variational autoencoder (VAE) to estimate item parameters with correlated and independent latent abilities, and we validate Q-matrix via the root mean square error (RMSE), bias, correlation, and AIC and BIC test score. The incorporation of a non-identity covariance matrix in a VAE requires a novel VAE architecture, which can be utilized in applications outside of education such as players performance evaluation, clinical trials assessment. Moreover, results show that the ML2P-VAE method is capable of estimating parameters and validating Q-matrix for models with a large number of latent variables with low computational cost, whereas traditional methods are infeasible for data with high-dimensional latent traits.

Keywords

Item Response Theory, Deep Generative Model, Interpretable Neural Network, Cognitive Diagnostic Model, Educational Assessment

1. INTRODUCTION

Item Response Theory (IRT) is a popular model for the understanding of human learning and problem-solving skill and to predict human behavior and performance. Since the 1950s [21], thousands of researchers have used IRT in fields, e.g., education, medicine, and psychology, and this includes many critical contexts such as survey analysis, popular questionnaires, medical diagnosis, and school system assessment.

More recently, computer-assisted open-access learning has gotten more popular worldwide, e.g., Khan Academy, Coursera, and EdX have developed a new challenge to handle large-scale student and trace performance [15].

M. Hasan, L. Y. Deng, J. Sabatini, D. Bowman, C.-C. Yang, and J. Hollander. Effect of q-matrix misspecification on variational autoencoders (VAE) for multidimensional item response theory (MIRT) models estimation. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 811–815, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853016>

In the deep learning domain, a revolution in deep generative models via variational autoencoders [12] [14], has demonstrated an impressive ability to perform fast inference for complex MIRT models. In this research, we present a novel application of variational autoencoders to MIRT, explore independent and correlated latent traits in the MIRT model via simulated data, and apply them to real-world examples. We then show the impact of Q-miss (wrong Q-matrix) when mixed compared with the original Q-matrix (Q-true).

Specifically, in this paper, we have explored two research questions as follows: first, how to use variational autoencoder in the estimation of MIRT models with large numbers of correlated and independent latent traits? Second, how are the effects of various factors such as the percentage of misfit items in the test and item quality (e.g., discrimination) on item and model fit in case of misspecification of Q-matrix?

Most closely related to the present work, Converse [2] utilized variational autoencoders (VAE) to estimate item parameters with correlated latent abilities and directly compared ML2P-VAE with traditional methods. Curi [1] introduces novel variational autoencoders to estimate item parameters with independent latent traits. Guo [16] explored the neural network approach and compared the outcome with the DINA model. Converse [3] compared outcomes between autoencoders (AE) and variational autoencoders (VAE). Wu [21] investigated the novel application of variational inference and incorporated IRT in the model via simulated and real data. Different from Converse [2], and Curri [1], we use both independent and latent traits in the VAE model. Moreover, we have explored the effect of Q-matrix misspecification in MIRT parameter estimation via different fit statistics, e.g., RMSE, BIAS, AIC, & BIC score measures.

The Multidimensional Logistic 2-Parameter (ML2P) model gives the probability of students answering a particular question as a continuous function of student ability [14]. There are two types of parameters associated with each item: a difficulty parameter b_i for the item i , and a discrimination parameter $a_{ik} \geq 0$ for each latent trait, k quantifying the hierarchy of ability k required to answer the item i correctly. The ML2P model gives the probability of a student j with latent abilities answering an item i correctly as

$$P(u_{ij} = 1 | \Theta_j; a_i, b_i) = \frac{1}{1 + \exp[-\sum_{k=1}^K a_{ik}\theta_{jk} + b_i]} \quad (1)$$

2. VARIATIONAL AUTOENCODERS

The variational autoencoder (VAE) is a directed model that uses learned approximate inference and can be trained purely with gradient-based methods [12]. It is similar to auto-encoders but with a probabilistic twist. VAE makes the additional assumption that the low-dimensional representation of data follows some probability distribution $N(0, I)$, and fits the encoded data to this distribution.

The main use of a VAE as a generative model, VAE generates X to \hat{X} after training and feed-forward through the decoder. By Bayes' rule, we can write the unknown posterior distribution. In our case, we generalized VAE as $N(\mu, \Sigma)$. In order to keep both P and Q distribution similar, Kullback-Liebler divergence $D_{KL}(P \parallel Q)$ plays a key role in the neural network loss function. The KL-Divergence is given by as follows:

$$KL[q(\Theta | x) \parallel f(\Theta | x)] = E_{\theta \sim q(\Theta | x)} [\log q(\Theta | x) - \log f(\Theta | x)] \quad (2)$$

As shown by Kingma and Welling [12] that minimizing Eq. 2 while still reconstructing input data is equivalent to maximizing.

$$E_{\theta \sim q(\theta | x)} [\log P(X=x|\Theta) - KL[q(\Theta|x) \parallel f(\Theta)]] \quad (3)$$

Next, the VAE is trained by a gradient descent algorithm to minimize the loss function. In this case, L_0 is the cross-entropy loss function and λ is a regularization hyperparameter.

$$L(W) = L_0(W) + \lambda KL[q(\Theta | x) \parallel f(\Theta)] \quad (4)$$

Root mean squared error (RMSE): RMSE criterion reflects the average magnitude of the bias between the true item parameters and their associated estimates. A smaller RMSE suggests higher estimation accuracy. Moreover, we also looked into Akaike information criterion (AIC), and Bayesian information criterion (BIC) score to explore MIRT model estimation when using Q-Miss.

First, we have incorporated the independent and correlated latent traits via the ML2P-VAE model proposed by Curi [1], and Converse [2]. We have extended this work by validating Q-matrix based on root, mean, square, error (RMSE), BIAS, and Correlation score.

We made modifications to the architecture of the neural network to allow for the interpretation of weights and biases in the decoder as item parameter estimates, and activation values in the encoded hidden layer as ability parameter estimates. As we know, sometimes researchers call neural networks are usually uninterpretable and function as a black-box model. However, following the addition of Q-matrix in the second from the last layer will make NN more interpretable.

The required modifications are as follows. The decoder of the variational autoencoder has no hidden layers. The non-zero weights in the decoder, connecting the encoded distribution to the output layer, are determined by a given

Q-matrix [19]. Thus, these two layers are not densely connected. The output layer must use the sigmoid activation function as follows:

$$\sigma(z_i) = \frac{1}{1 + e^{-z_i}} \quad (5)$$

When latent traits are assumed to be correlated, a full correlation matrix must be provided for the ML2P-VAE model. However, a correlation matrix is not required when latent traits are assumed to be independent. This corresponds to the fixed covariance matrix Σ_1 . ML2P-VAE can estimate ability, discrimination, and difficulty parameters, but it does not estimate correlations between latent traits.

Also, the input to our neural network consists of n nodes, representing items on an assessment. After a sufficient number of hidden layers of sufficient size, the encoder outputs $K + K(K + 1)/2$ nodes. The architecture for correlated latent traits is more complex than we think (See Visualization of Deep-VAE architecture for two correlated latent traits and ten input items model via this link [tinyurl.com/aied22]).

2.1 Q-Matrix and Misspecification of Q-matrix

Specification of Q-matrix is mainly criticized because of its subjective nature [17]. Misspecification in the cognitive diagnostic model (CDM) mostly occurs because of the types of the attributes, construct of the attribute, Q-matrix, or selected cognitive diagnostic model [6]. In this experiment, we utilized only a misfit source because Q-matrix misspecification was examined, and no changes were made in students' responses. In the study, the Q-matrix was misspecified by a mixed approach, and misfit items used in this study are presented in Table 1 (See misfit items table in Appendix 2: tinyurl.com/aied22). When the Q-matrix was misspecified, one attribute was translated from 1 to 0, and another attribute was translated from 0 to 1, but the number of measured attributes didn't change, which is referred to as mixed.

In the architecture of the model ML2P-VAE, we train the neural network with the ADAM optimizer (pure stochastic gradient descent). A simulated assessment with six latent abilities used two hidden layers of sizes 50 and 25. The largest network we used was for an assessment of 20 latent abilities, which utilized two hidden layers of sizes 100 and 50.

3. THE DEEP-Q ALGORITHM

For convenience, we are calling this algorithm the Deep-Q algorithm. The steps of the Deep-Q algorithm are as follows-

- Step 1:** Use the variational autoencoder and multidimensional item response theory (ML2P-VAE) model [2] to estimate students' ability and item parameters based on Q-True and the response data.
- Step 2:** Compute all items' via RMSE, BIAS, and Correlation test score values based on Q-True and the student's ability and item parameters estimated at Step 1. We also use AIC and BIC scores to compare Q-true and Q-miss.
- Step 3:** Randomly misspecify Q-true by 10% and 20% to change Q-True.

Table 1: Q-matrix validation measures via RMSE, BIAS and Correlation score for discrimination (a), difficulty (b), and ability θ parameters with correlated latent traits

Data	Miss	Method	a.RMSE	a.BIAS	a.Corr	b.RMSE	b.BIAS	b.Corr	θ .RMSE	θ .BIAS	θ .Corr
		Q_{True}	0.1465	0.0100	0.9427	0.0750	0.0100	0.9988	0.8120	0.0476	0.5815
N=18000	10% $_{Miss}$	Q_{Miss}	0.2297	0.0195	0.8562	0.0993	0.0083	0.9986	0.8398	0.0637	0.5486
	20% $_{Miss}$	Q_{Miss}	0.2621	0.0466	0.7984	0.1684	0.0268	0.9962	0.8684	0.1660	0.5351
		Q_{True}	0.1007	0.0259	0.9094	0.2098	-0.0186	0.9984	0.5614	-0.0013	0.8686
N=60000	10% $_{Miss}$	Q_{Miss}	0.2525	0.0654	0.8430	0.2881	0.0129	0.9926	0.7288	0.0037	0.6859
	20% $_{Miss}$	Q_{Miss}	0.2200	0.0367	0.8777	0.2191	0.0243	0.9952	0.6561	0.0166	0.7551

Step 4: Repeat step-1 with Q-miss (Q-miss, Step 3).

Step 5: Compare Q-True(top row, boldface) with Q-miss. Q-true should yield a small RMSE/BIAS and a strong correlation score, AIC, and BIC score for difficulty, discrimination, and ability parameters.

4. METHODOLOGY

We ran experiments on two data sets: (i) 6 traits, 35 items, and 18000 students, and (ii) 20 traits, 200 items, and 60,000 students. It is also important to mention here that true parameter values, for both students and items, are only available for simulated data. When simulating data, we used the Python's SciPy package to generate a symmetric positive definite matrix with 1s on the diagonal (correlation matrix) and all matrix entries non-negative. All latent traits had correlation values between 0,1. We assumed that each latent trait was mean-centered at 0. Then, we sampled ability vectors to create simulated students. We generated a random Q-matrix where each entry $q_{ij} \sim \text{Bern}(0.2)$. If a column for each element after sampling from this Bernoulli distribution, one random element was changed to a 1. Discrimination parameters were sampled from a range so from 0.25 to 1.75 for each item i , and difficulty parameters were sampled uniformly from - 3 to 3. Finally, response sets for each student were sampled from the ML2P model using these parameters.

5. RESULTS

All experiments were conducted using TensorFlow for R and the ML2Pvae package [4] on an iMac computer with a 3.1 GHz Intel Core i5 via Google Colab Premium, 12 GB NVIDIA Tesla K80 GPU.

Table 1 presents the estimation accuracy of Q under Q-True and Q-miss. The range of values for each criterion is provided in the second and third row of Table 1, and the numbers in bold denote better performance in the associated criterion for the corresponding method, e.g., Q-Miss.

Overall, Table-1 indicates that the Deep-Q method yields a better fit statistic score and strong correlation score than the Q-miss situation when using a wrong Q-matrices. This result is corroborated by the correlation plots between the

true discrimination parameters and the weights of the decoder, displayed in Fig. 1 and 2 (see Appendix for larger view).

In addition, Q-matrix validation measures via AIC, BIC score for discrimination (a), difficulty (b), and ability θ parameters with correlated latent traits remain consistent with Table-1 outcome (see AIC and BIC scores in the Appendix).

In Fig 1(A and B), the correlation plots of discrimination parameter estimate for data with items and latent traits. Each color represents discrimination parameters relating to one of each latent skill. In the ability parameter, each color in the plot represents discrimination and ability parameters associated with each latent trait. Difficulty parameters are on the item level, not the latent trait level. So in each item, I have exactly one difficulty parameter b_i , regardless of the number of latent skills. The interpretation is similar for independent latent traits, as described in figure 1(A). Plots show correlated latent traits and show better outcomes compared to independent latent traits.

6. DISCUSSION AND CONCLUSION

An incorrect Q-matrix can lead to a significant change in the assessment outcomes when applied to CDMs. As a result, a Q-matrix validation strategy to reduce assessment error is becoming increasingly important. Several approaches, including EM-based and non-parametric methods, have shown the ability to identify and create an acceptable Q-matrix. However, to the best of the authors' knowledge, their experiment utilizes traditional IRT parameter estimation where they utilize low-dimensional latent traits and students' responses. However, the Deep-Q algorithm is most useful with high and low-dimensional data.

Moreover, Converse's [2] study shows that MIRT parameter estimation results via the ML2P-VAE model are competitive compared to traditional IRT parameter estimation methods. Our study used a Deep-Q algorithm, a deep learning-based algorithm, to identify and validate a Q-matrix for small and large-scale latent traits. Deep-Q could be useful for large-scale assessments, e.g., PISA and TIMSS.

ML2P-VAE is a novel technique that allows IRT parameter estimation of independent and correlated low and high-

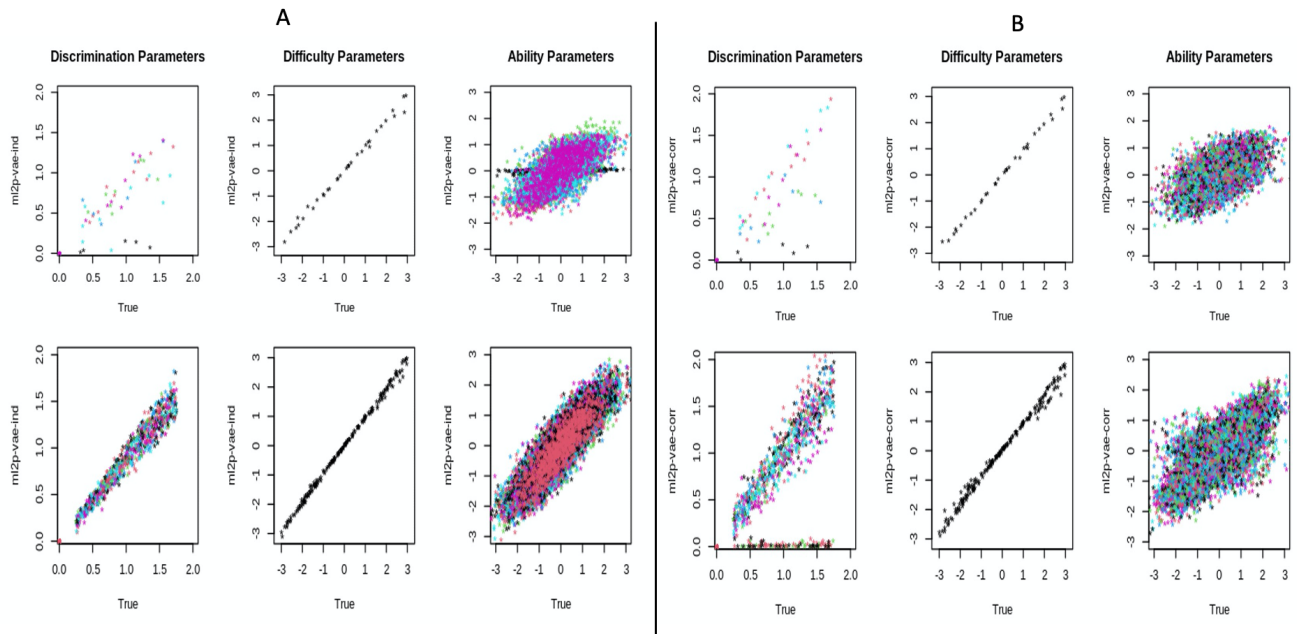


Figure 1: (A) Discrimination, Difficulty, and Ability Parameter Estimates with Independent Latent Traits (B) Discrimination, Difficulty, and Ability Parameter Estimates with Correlated Latent Traits.

dimensional latent traits. Ultimately, it can be said that the Deep-Q algorithm succeeds in detecting misfit items in both large and small sample cases. ML2P-VAE methods and Deep-Q are most useful on high-dimensional data, but even when applied to smaller data sets where traditional techniques are feasible, the results from current methods are competitive.

7. ACKNOWLEDGMENTS

This research was sponsored by the National Science Foundation under the award The Learner Data Institute (bit.ly/36Bi93m) (award #1934745). The opinions, findings, and results are solely the authors' and do not reflect those of the funding agencies. Thanks to Geoff Converse, Andrew Ott, and LDI team for their suggestions and comments.

8. REFERENCES

- [1] Curi, M., Converse, G. A., Hajewski, J., Oliveira, S.: Interpretable variational autoencoders for cognitive models. 2019 International Joint Conference on Neural Networks (IJCNN) **1-8**. IEEE.(2019).DOI: 10.1109/IJCNN.2019.8852333
- [2] Converse, G., Curi, M., Oliveira, S., Templin, J.: Estimation of multidimensional item response theory models with correlated latent variables using variational autoencoders. Machine Learning, **1-18**.(2021). DOI:10.1007/s10994-021-06005-7
- [3] Converse, G., Curi, M., Oliveira, S.: Autoencoders for educational assessment. International Conference on Artificial Intelligence in Education.**41-45**.Springer.(2019).<https://doi.org/10.1007/978-3-030-23207-8-8>
- [4] Converse, G.: ML2Pvae: VAE models for IRT parameter estimation.(2020) <https://CRAN.R-project.org/package=ML2Pvae>, r package version1.0.0.
- [5] Liu, C. W., Chalmers, R. P.: Fitting item response unfolding models to likert scale data using mirt in R. PloS one. **13(5)**.(2018). <https://doi.org/10.1371/journal.pone.0196292>
- [6] Chen, J., de la Torre, J., Zhang, Z.: Relative and absolute fit evaluation in cognitive diagnosis modeling. Journal of Educational Measurement, **50(2)**,123-140.(2013).<https://doi.org/10.1111/j.1745-3984.2012.00185>.
- [7] Kingma, D. P., Welling., M.: Auto-encoding variational bayes. (2013). <https://doi.org/10.48550/arXiv.1312.6114>
- [8] Rezende, D. J., Mohamed, S. and Wierstra, D.: Stochastic backpropagation and approximate inference in deep generative models. (2014).<https://doi.org/10.48550/arXiv.1401.4082>
- [9] Harris, D.: Comparison of 1-, 2-, and 3-parameter IRT models. Educational Measurement: Issues and Practice,**8(1)**, 35-41. (1989).<https://doi.org/10.1111/j.1745-3992.1989.tb00313.x>
- [10] Leighton, J. P., Gierl, M. J., Hunka,S. M.: The attribute hierarchy method for cognitive assessment: A variation on Tatsuoaka's rule-space approach. J. Educ.

- Meas., vol. 41, **205–237**,(2004).
<https://doi.org/10.1111/j.1745-3984.2004.tb01163.x>
- [11] Torre, J. de la. and Chiu, C. Y.,: "A General Method of Empirical Q-matrix Validation," *Psychometrika*, vol. 81, no. 2, **253–273**.(2016). DOI: 10.1007/s11336-015-9467-8
- [12] Kingma, D. P., & Welling, M.: Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114.(2013).
<https://doi.org/10.48550/arXiv.1312.6114>
- [13] DeCarlo, L. T.: On the analysis of fraction subtraction data: The DINA model, classification, latent class sizes, and the q-matrix. *Appl. Psychol. Meas.*, vol. 35, no. 1, **8–26**,(2011).<https://doi.org/10.1177/0146621610377081>
- [14] McKinley, R., Reckase, M.: The use of the general Rasch model with multidimensional item response data. American College Testing. (1980)
- [15] Piech C., Bassen J., Huang J., Ganguli S., Sahami M., Guibas L.J., Sohl-Dickstein J.: Deep knowledge tracing. *Advances in neural information processing systems*. **28**.(2015)
- [16] Guo, Q. Cutumisu, M., Cui, Y.: A neural network approach to estimate student skill mastery in cognitive diagnostic assessments. (2017).<https://doi.org/10.7939/R35H7C71D>
- [17] Rupp, A. A., Templin, J.: The effects of Q-matrix misspecification on parameter estimates and classification accuracy in the DINA model. *Educational and Psychological Measurement*, 68(1), 78-96.(2008).<https://doi.org/10.1177/0013164407301545>
- [18] Rezende, D. J., Mohamed, S., Wierstra, D.: Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning* **1278–1286**. PMLR. (2014).<https://doi.org/10.48550/arXiv.1401.4082>
- [19] Tatsuoka, K.K.: Rule space: an approach for dealing with misconceptions based on item response theory. *J. Educ. Meas.* **20(4)**, 345–354 (1983)
- [20] Ma, W., Torre, J. de la.: An empirical Q-matrix validation method for the sequential generalized DINA model. *Br. J. Math. Stat. Psychol.*, (2019).
<https://doi.org/10.1111/bmsp.12156>
- [21] Wu, M., Davis, R. L., Domingue, B. W., Piech, C., Goodman, N.: Variational item response theory: Fast, accurate, and expressive. arXiv preprint arXiv:2002.00276.(2020).<https://doi.org/10.48550/arXiv.2002.00276>
- [22] Zhang, J., Shi, X., King, I., Yeung, D. Y.: Dynamic key-value memory networks for knowledge tracing. In: 26th International world wide web conference (WWW 2017) **765–774**.(2017).
<https://doi.org/10.1145/3038912.3052580>

9. APPENDIX

Please follow this link for Appendixes: tinyurl.com/aied22

Towards Personalised Learning of Psychomotor Skills with Data Mining

Miguel Portaz^{*}
Computer Science School, UNED
mportaz1@alumo.uned.es

Olga C. Santos[†]
Computer Science School, UNED
ocsantos@dia.uned.es

ABSTRACT

Data Mining (DM) currently represents a key element for improving the acquisition of knowledge and for providing proper feedback in educational environments. In this field, as well as in others, psychomotor skills represent a unique way to advance in the knowledge of human behaviours. Our research apply in two equally promising, but different domains. On one hand, we would be able to improve the learning of psychomotor skills at educational level with the use of different DM techniques. This may includes learning martial arts or supporting the acquisition of locomotor abilities. On the other hand, we would like to expand our DM research far beyond the basis of the aforementioned educational field. Thus, we can evaluate other users, such patients, with the aim of improving the re-learning of motor capabilities during recovery processes on rehabilitation, or even to detect cognitive impairments, analysing slight psychomotor alteration: at early stages using DM. The latter includes gait analysis, which are currently used for screening, but not so much for predicting purposes. Although our research is still at early stages, we are following the principles set on previous researches, such those included in our intelligent Expertise Level Assessment (iELA) method.

Keywords

Educational Data Mining, Personalised Learning, Psychomotor Skills, Active Ageing and Education

1. INTRODUCTION

In order to properly apply state-of-the-art DM techniques for personalised learning, we have developed the following taxonomy, distinguishing between *mutually-exclusive* and *non-mutually-exclusive* movements [31]. This differentiation is particularly relevant for psychomotor activity analysis, since learners should be modelled according subtle details, which

^{*}1st year PhD Candidate.

[†]aDeNu Research group

M. Portaz and O. C. Santos. Towards personalised learning of psychomotor skills with data mining. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 816–820, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853071>

are not often obvious. In the first *mutually-exclusive* group we can find most of the Human Activity Recognition (HAR) systems, which basically use DM and Machine Learning (ML) techniques to analyse human movements, that cannot be executed at the same time (running, walking, sleeping, etc). In our case, it is extremely pertinent to be able to analyse *non-mutually-exclusive* movements, meaning distinguish different behaviours which are similar. This is the case of analysing distinct personal gait registries or matching several gaits gathered from different users or patients. For recognising the specific *non-mutually-exclusive* human movements, we can use inertial sensors, where the information is often collected in form of Time-Series (TS), like the ones depicted in Fig. 1.

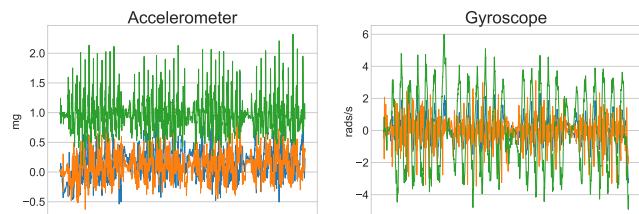


Figure 1: Inertial data recorded representing human activity. On the left side we have the three axis (x, y, z) gathered using an accelerometer. On the right side we have the three axis (x, y, z) gathered using a gyroscope [31].

Fig. 2 depicts another method suitable for breaking down *non-mutually-exclusive* human movements, in this case utilising AI-based video analytics.

Many other different controllers may be used for the purpose of gathering information related to *mutually-exclusive* and *non-mutually-exclusive* human movements, such Microsoft Kinect sensor [29] or LIDAR¹ sensor [20]. Whether the method and sensors used, the aim of the research disclosed on this paper is to analyse, with different DM techniques, several data sets gathered from the execution of human movements, to anticipate psychomotor learning behaviours supporting appropriate feedback or analysis.

This research may also have a specific impact on achieving well-being through active ageing, since the correct execution of certain movements at certain ages allows us to take care

¹Acronym of LIght Detection And Ranging.



Figure 2: Multi-person Pose Estimation with OpenPose [30].

of ourselves properly. Therefore, the use of DM techniques, which allow us to automatically assess whether we perform certain motor movements correctly or if we preserve certain psychomotor skills, is essential for our comfort and well-being.

2. RELATED RESEARCH

The first steps for our research relies on the iELA method disclosed in [31], which uses *quaternions*, a specific group of hyper-complex numbers, to fuse the transformed inertial data, gathered during the practice of different martial arts. iELA uses DM to provide an innovative method to infer performance level, showing its relevance assisting psychomotor learning and educational systems. iELA follows the principles set in [34], which details how the use of different sensors, together with a specific methodology, to support psychomotor learning processes in educational environments. These foundations are also used in [35, 12, 9, 18] since motor tasks can be consolidated into memory through repetitions, creating a long-term neuromuscular memory for specific tasks.

Other intelligent tutoring system related with psychomotor skills can be found in [24], although in this case the authors use physiological sensors. Regarding video analysis, [28] introduce the design, development, deployment and evaluation of video games to support locomotor acquisition in a classroom setting. Other examples are related with the learning of playing piano [8], practising martial arts [33], performing dance step movements [33, 16] or improving sport technique [7], among others.

Some researches, as in [25, 37, 36, 15], use video analysis techniques for human pose estimation, in contrast, iELA foundations analyse inertial data gathered with the use of IMU² sensors, as in [22, 32, 13, 5, 21]. iELA analyses the data collected in form of TS, as in [40] but using two different approaches: extracting features, as in [3, 4], and using Convolutional Neural Networks (CNNs), as in [19, 14], taking into account that TS have practically the same topology than images, as mentioned in [6]. Other researches use another DM techniques [23], machine learning methods [26] or neural networks [39, 2, 11].

The use of various methods to estimate accurately the hu-

²Acronym of Inertial Measurement Unit

man pose is also common, as in [27, 38, 12] which use two different types of data, the one provided by the IMU and the one gathered from video footage.

In addition to the analysis of psychomotor activities in Educational DM systems, the detection and analysis of the human pose in other research areas, can also take advantage of the different DM techniques described above. Thus, [1] provides automated movement assessment for stroke rehabilitation using video analysis. At this point it is important to denote how pre-dementia stages (motor cognitive risk syndrome) are characterised in some cases by slow gait, as disclosed in [10]. Consequently, this research has an impact on other areas related to psychomotor aspects and, therefore, could also provide tools to promote innovation in active and healthy ageing, increasing healthy life expectancy³.

3. EXPECTED CONTRIBUTION

In aspects exclusively related to the detection of the human pose in educational environments, in addition to iELA, neither of the examples described above, evaluate the level of performance. As iELA is setting up the principles for evaluating the level of performance of complex movements in martial arts, our aim is to extrapolate iELA to other domains, including educational or health behavioural. The use of *quaternions* to fuse inertial information, with the addition of other sources, such as those provided by video footage, may provide innovative mechanisms on educational and non educational environments that require the analysis of psychomotor skills.

Although one of the desired goals is to merge information coming from different sensors, including video footage, we carefully need to deal with one of the main drawbacks of using a video-based approach for human activity classification purposes, which are the issues in terms of privacy [17]. This will be particularly relevant for fulfilling IDEA (inclusion, diversity, equity, and accessibility) approaches, as well as for satisfying the guidelines for a trustworthy AI, including transparency, explainability, fairness, robustness and the aforementioned privacy.

On this research, we will also consider the analysis of the affective state of the participants within data gathering and in relation to the execution of the movements under our study.

4. RESEARCH QUESTIONS

The affirmative answer to the iELA research question carried out in the Master Thesis [31]:

RQ-1 *Can we use these DM driven analysis (iELA) to automatically classify practitioners according their expertise level?*

represents the starting point for defining the next research question to be carried out in the on going Doctoral Thesis:

RQ-2 *May we use DM approaches to assess the level of expertise accomplished while performing motor activities on*

³<https://ec.europa.eu/social/main.jsp?langId=en&catId=1062>

different psychomotor learning schemes (from martial arts techniques to physical movements to benefit active ageing) thus achieving a domain transfer?

which will be the main objective of this research.

5. PROPOSED METHODOLOGY

Our proposed methodology is based on iELA, a DM method for an intelligent expertise level assessment which analyses high volumes of inertial data. In this case, when breaking down inertial data, without fusing it into *quaternions*, we cannot clearly distinguish whether the representation corresponds to an expert user, which may be extrapolated to a patient without a pathology, who performs certain motor activity, as depicted in Fig. 3. When fusing this information into *quaternions*, see Fig. 4, we can clearly have information about how the motor activity was performed. On the other hand, if we analyse the same movement executed by a beginner user, which may be also extrapolated to a patient with a pathology, then we can obtain a different depiction, as in Fig. 5.

This approach is based in analysing quality datasets gathered while executing human psychomotor activities. The DM analysis of this data will follow iELA approaches, and may include the use of CNNs and the examination of the extracted features from TS.

6. CONCLUSIONS

Research in the analysis of psychomotor skills represents an important challenge, overall when we are going far beyond the distinction of the *mutually exclusive* movements disclosed alongside this article. At this stage we are looking forward for feedback to support our research with the aim of demonstrating how our iELA method provides a top-notch approach in the treatment and analysis of different psychomotor behaviours on educational, learning or even on health scenarios.

The use of the DM techniques developed in this research may have different fields of application at educational level, but also can be useful for achieving the benefits of active ageing.

7. ACKNOWLEDGMENTS

The work reported is framed in the project "INTELLIGENT INTRA-SUBJECT DEVELOPMENT APPROACH TO IMPROVE ACTIONS IN AFFECT-AWARE ADAPTIVE EDUCATIONAL SYSTEMS" (INT2AFF) led by Dr. Santos at UNED and funded by the Spanish Ministry of Science, Innovation and Universities, the Spanish Agency of Research and the European Regional Development Fund (ERDF) under Grant PGC2018-102279-B-I00, specifically within the scenario "SC4-motor skills", which focuses on analysing the influence of emotions in motor skills learning.

8. REFERENCES

- [1] T. Ahmed, K. Thopalli, T. Rikakis, P. Turaga, A. Kelliher, J.-B. Huang, and S. L. Wolf. Automated movement assessment in stroke rehabilitation. *Frontiers in Neurology* 12 (agosto): 720650, 2021.
- [2] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz. Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine. In J. Bravo, R. Hervás, and M. Rodríguez, editors, *Ambient Assisted Living and Home Care*, pages 216–223, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.
- [3] A. Avci, S. Bosch, M. Marin-Perianu, R. Marin-Perianu, and P. Havinga. Activity recognition using inertial sensing for healthcare, wellbeing and sports applications: A survey. page 11, 2010.
- [4] M. Barandas, D. Folgado, L. Fernandes, S. Santos, M. Abreu, P. Bota, H. Liu, T. Schultz, and H. Gamboa. TSFEL: Time series feature extraction library. 11:100456, 2020.
- [5] A. Bayat, M. Pomplun, and D. A. Tran. A study on human activity recognition using accelerometer data from smartphones. 34:450–457, 2014.
- [6] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013.
- [7] V. Camomilla, E. Bergamini, S. Fantozzi, and G. Vannozzi. Trends supporting the in-field use of wearable inertial sensors for sport performance evaluation: A systematic review. 18(3):873, 2018.
- [8] B. Caramiaux, F. Bevilacqua, C. Palmer, and M. Wanderley. Individuality in piano performance depends on skill learning. In *Proceedings of the 4th International Conference on Movement Computing*, pages 1–7. ACM, 2017.
- [9] A. Casas-Ortiz. Capturing, modelling, analyzing and providing feedback in martial arts with artificial intelligence to support psychomotor learning activities. 2020. Retrieved June 15, 2021, from <http://www.ia.uned.es/docencia/posgrado/master-tfm-archivo.html>.
- [10] Y. Cedervall, A. M. Stenberg, H. B. Åhman, V. Giedraitis, F. Tinmark, L. Berglund, K. Halvorsen, M. Ingelsson, R. Erik, and A. C. Aberg. Timed up-and-go dual-task testing in the assessment of cognitive function: A mixed methods observational study for development of the uddgait protocol. *International Journal of Environmental Research and Public Health* 17 (5): 1715, 2020.
- [11] A. Choi, H. Jung, and J. H. Mun. Single inertial sensor-based neural networks to estimate COM-COP inclination angle during walking. 19(13):2974.
- [12] A. Corbí and O. C. Santos. MyShikko: Modelling knee walking in aikido practice. In *Adjunct Publication of the 26th Conference on User Modeling, Adaptation and Personalization*, pages 217–218. ACM, 2018.
- [13] O. Dehzangi and V. Sahu. IMU-based robust human activity recognition using feature analysis, extraction, and reduction. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 1402–1407. IEEE, 2018.
- [14] A. Dempster, F. Petitjean, and G. I. Webb. ROCKET: exceptionally fast and accurate time series classification using random convolutional kernels. 34(5):1454–1495, 2020.

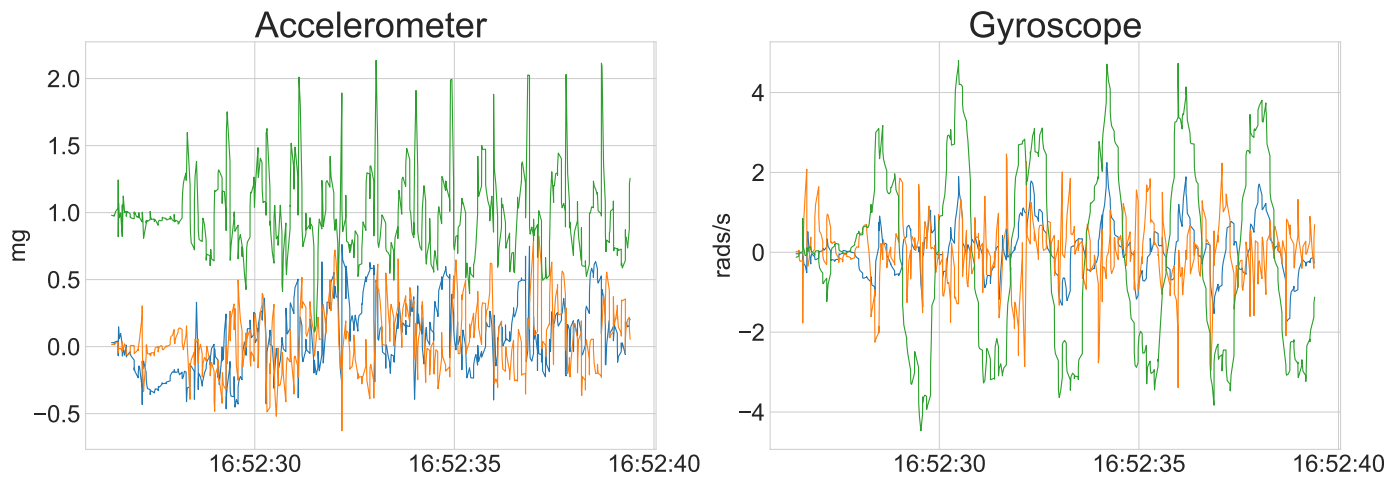


Figure 3: Signals corresponding to the raw inertial data gathered during the execution of a martial art movement in the case of an *expert* practitioner [31].

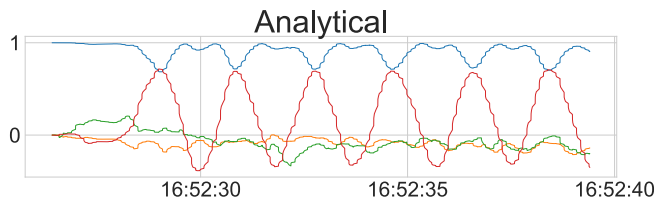


Figure 4: Signals corresponding to *quaternion* fusion obtained during the same movement represented in Fig. 3 and performed by the same user. Analytical means the method used for the *quaternion* fusion [31].

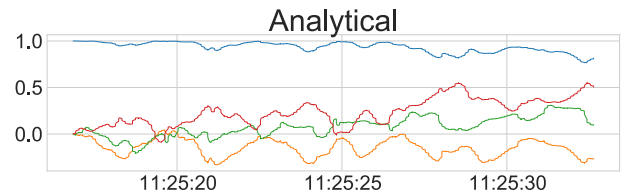


Figure 5: Signals corresponding to *quaternion* fusion obtained during the same movement represented in Fig. 4 but performed by another user. Analytical means the method used for the *quaternion* fusion [31].

- [15] F. Demrozi, G. Pravadelli, A. Bihorac, and P. Rashidi. Human activity recognition using inertial, physiological and environmental sensors: a comprehensive survey. 8:210816–210836, 2020.
- [16] A. Dias Pereira dos Santos, K. Yacef, and R. Martinez-Maldonado. Let’s dance: How to build a user model for dance students using wearable technology. In *Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization, UMAP ’17*, page 183–191, New York, NY, USA, 2017. Association for Computing Machinery.
- [17] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, T. Darrell, and K. Saenko. Long-term recurrent convolutional networks for visual recognition and description. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2625–2634, 2015.
- [18] J. Echeverria and O. C. Santos. Kunitron: A multimodal psychomotor intelligent learning system to provide personalized support when training karate combats. 2021. First International Workshop on Multimodal Artificial Intelligence in Education (MAIED 2021) held in conjunction with the AIED 2021. To be published in CEUR.
- [19] H. I. Fawaz, B. Lucas, G. Forestier, C. Pelletier, D. F. Schmidt, J. Weber, G. I. Webb, L. Idoumghar, P.-A. Muller, and F. Petitjean. InceptionTime: Finding AlexNet for time series classification. 34(6):1936–1962, 2020.
- [20] M. Fürst, G. Shriya T. P., R. Schuster, O. Wasenmüller, and D. Stricker. Hperl: 3d human pose estimation from rgb and lidar. 2020.
- [21] N. Golestani and M. Moghaddam. Human activity recognition using magnetic induction-based motion signals and deep recurrent neural networks. *Nature communications*, 11(1):1551, March 2020.
- [22] E. A. Heinz, K. S. Kunze, M. Gruber, D. Bannach, and P. Lukowicz. Using wearable sensors for real-time recognition tasks in games of martial arts - an initial experiment. In *2006 IEEE Symposium on Computational Intelligence and Games*, pages 98–102. IEEE, 2006.
- [23] C. Jobanputra, J. Bavishi, and N. Doshi. Human activity recognition: A survey. 155:698–703, 2019.
- [24] J. W. Kim, C. Dancy, and R. A. Sottolare. Towards using a physio-cognitive model in tutoring for psychomotor tasks. page 14, 2018. Retrieved May 10, 2021, from <https://digitalcommons.bucknell.edu/cgi/>

- viewcontent.cgi?article=1055context=fac_conf.
- [25] J. Lee and H. Jung. TUHAD: Taekwondo unit technique human action dataset with key frame-based CNN action recognition. 20(17):4871, 2020.
- [26] A. Mannini and A. M. Sabatini. Machine learning methods for classifying human physical activity from on-body accelerometers. 10(2):1154–1175, 2010.
- [27] T. v. Marcard, G. Pons-Moll, and B. Rosenhahn. Human pose estimation from video and IMUs. 38(8):1533–1547, 2016.
- [28] J. McGann, J. Issartel, L. Hederman, and O. Conlan. Hop.skip.jump.games: The effect of “principled” exergameplay on children’s locomotor skill acquisition. *British Journal of Educational Technology*, 51, 11 2019.
- [29] S. Obdrzalek, G. Kurillo, F. Ofli, R. Bajcsy, E. Seto, H. Jimison, and M. Pavel. Accuracy and robustness of kinect pose estimation in the context of coaching of elderly population. *Conference proceedings : ... Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Conference*, 2012:1188–93, 08 2012.
- [30] Y. Okugawa, M. Kubo, H. Sato, and V. Bui. Evaluation for the synchronization of the parade with openpose. *Proceedings of International Conference on Artificial Life and Robotics*, 24:443–446, 01 2019.
- [31] M. Portaz, A. Corbi, A. Casas-Ortiz, and O. C. Santos. Intelligent expertise level assessment using wearables for psychomotor learning in martial arts (master’s thesis). *National Distance Education University (UNED)*, 06 2021. Retrieved Apr 15, 2022, from <http://www.ia.uned.es/docencia/posgrado/master-tfm-archivo.html>.
- [32] S. M. Rispens, M. Pijnappels, K. S. van Schooten, P. J. Beek, A. Daffertshofer, and J. H. van Dieën. Consistency of gait characteristics as determined from acceleration data collected at different trunk locations. *Gait Posture*, 40(1):187–192, 2014.
- [33] O. C. Santos. Towards personalized vibrotactile support for learning aikido. In Lavoué, H. Drachsler, K. Verbert, J. Broisin, and M. Pérez-Sanagustín, editors, *Data Driven Approaches in Digital Education*, volume 10474, pages 593–597. Springer International Publishing. Series Title: Lecture Notes in Computer Science.
- [34] O. C. Santos. Training the body: The potential of AIED to support personalized motor skills learning. 26(2):730–755, 2016.
- [35] O. C. Santos and M. H. Eddy. Modeling psychomotor activity: Current approaches and open issues. In *Adjunct Publication of the 25th Conference on User Modeling, Adaptation and Personalization, UMAP ’17*, page 305–310, New York, NY, USA, 2017. Association for Computing Machinery.
- [36] K. Sun, B. Xiao, D. Liu, and J. Wang. Deep high-resolution representation learning for human pose estimation. In *CVPR*, page 11, 2019.
- [37] A. Toshev and C. Szegedy. DeepPose: Human pose estimation via deep neural networks. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1653–1660. IEEE, 2014.
- [38] T. von Marcard, B. Rosenhahn, M. J. Black, and G. Pons-Moll. Sparse inertial poser: Automatic 3d human pose estimation from sparse IMUs. 2017.
- [39] S. Wan, L. Qi, X. Xu, C. Tong, and Z. Gu. Deep learning models for real-time human activity recognition with smartphones. 25(2):743–755, 2020.
- [40] H. Zhou, H. Hu, and Y. Tao. Inertial measurements of upper limb motion. 44(6):479–487, 2006.

Using a Randomized Experiment to Compare the Performance of Two Adaptive Assessment Engines

Jeffrey Matayoshi
McGraw Hill ALEKS
jeffrey.matayoshi
@mheducation.com

Hasan Uzun
McGraw Hill ALEKS
hasan.uzun
@mheducation.com

Eric Cosyn
McGraw Hill ALEKS
eric.cosyn
@mheducation.com

ABSTRACT

Knowledge space theory (KST) is a mathematical framework for modeling and assessing student knowledge. While KST has successfully served as the foundation of several learning systems, recent advancements in machine learning provide an opportunity to improve on purely KST-based approaches to assessing student knowledge. As such, in this work we compare the performance of an existing KST-based adaptive assessment to that of a newly developed version—with this new version combining the predictive power of a neural network model with the strengths of existing KST-based approaches. Using a cluster randomized experiment containing data from approximately 140,000 assessments, we show that the new neural network assessment engine improves on the performance of the existing KST version, both on standard classification metrics, as well as on measures more specific to the student learning experience.

Keywords

Adaptive assessment, neural networks, knowledge space theory

1. INTRODUCTION

Combining elements of probability theory and combinatorics, knowledge space theory (KST) is a mathematical approach to the modeling and assessing of student knowledge [12, 15]. KST-based adaptive assessments focus on identifying the set of topics that are most likely to be known by a student, with many such implementations having been developed over the years [6, 7, 18, 20]. The particular system at the center of this work is ALEKS, an adaptive learning system covering subjects such as math, statistics, and chemistry. The foundation of the system is an *initial assessment* that is given to a student at the beginning of an ALEKS course, with the information from this assessment being used to guide their subsequent learning in the system. The current version of the initial assessment engine uses a KST-based model that employs many features and concepts described in the

KST literature (informative summaries of these ideas can be found in [13] and Chapter 13 of [15]).

Several works have evaluated the validity and reliability of KST-based assessments, with the results indicating that such assessments are both accurate and valid [5, 6, 10, 28]. However, current innovations in machine learning and, in particular, neural network models provide an opportunity to improve on purely KST-based approaches to adaptive assessments. As such, a newer version of the ALEKS initial assessment was recently developed and released into production for testing. By augmenting the KST-based adaptive assessment approach with a neural network model that can better leverage the years of accumulated ALEKS user data, we hope to make improvements to the accuracy and efficiency of the initial assessment. Thus, in the current study we evaluate this new assessment engine by analyzing the results from a randomized experiment—or, A/B test—comparing the performance of the two different versions of the ALEKS initial assessment.

2. BACKGROUND

In this section, we give a brief background of the ALEKS system and the two versions of its initial assessment engine. Within the ALEKS system, a *topic* is a problem type that covers a discrete unit of an academic course—Figure 1 contains a screen capture of an example math topic titled “Introduction to solving an equation with parentheses.” A *knowledge state* is a collection of topics that, conceivably, a student can know at any one time. The collection of knowledge states is known as the *knowledge space*. Based on the knowledge space, the topics in an ALEKS course contain many *prerequisite* relationships. That is, topic *a* is a prerequisite for topic *b* if *a* contains certain core material that must be mastered before learning the material in *b*.

In order to ensure students are learning the most appropriate topics, the initial assessment is given at the start of an ALEKS course, with the purpose of this assessment being to measure the student’s incoming knowledge. In this assessment, a student is presented topics from the course, and for each topic they can either submit a response—which is then graded as either correct or incorrect—or they can click on the “I don’t know” button if they are unable to answer the question. This assessment is adaptive, in that it selects the current question based on the responses to the earlier questions in the assessment. At most 30 questions are asked, which balances the need to acquire information on the stu-

J. Matayoshi, H. Uzun, and E. Cosyn. Using a randomized experiment to compare the performance of two adaptive assessment engines. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 821–827, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853189>

dent’s knowledge state with the possibility of overwhelming the student.¹ After each question, a probability is computed for each topic in the course, with this probability estimating how likely it is that the student knows the topic. At the end of the assessment, based on both these probability estimates and the prerequisite relationships in the knowledge space, the ALEKS system partitions the topics in the course into the following three categories.

- Topics that are most likely known
- Topics that are most likely unknown
- All remaining topics (uncertain)

The student’s knowledge state consists of the topics in the known category.² Due to the enormous numbers of possible knowledge states, KST-based assessments typically use simplifications in order to model the relationships between topics. Perhaps the most common approach is to focus solely on identifying the prerequisite—or, topic to topic—relationships when developing the knowledge space; these are variously known as partial order or ordinal knowledge spaces [8, 9, 11, 15]. While adaptive assessments using these approaches have been successful [6, 7, 10, 20], there is an inherent loss of information when mainly focusing on the pairwise relationships between the topics. Although methods have been outlined for moving beyond pairwise relationships and looking at larger groups of topics simultaneously, the computational complexity grows substantially with each added topic, and research has thus far been mostly limited to small groups of at most three or four topics [11].

Motivated by this issue, the new version of the ALEKS assessment engine is powered by a neural network model that, for each topic, estimates the probability of a correct answer. The advantage of a neural network is that it has the flexibility to model more complex relationships between the topics—that is, it can go beyond focusing on specific relationships between pairs or very small groups of topics. However, for both practical and theoretical reasons, it’s desirable that the neural network model work within the existing KST framework of the ALEKS system. While other attempts have been made at applying neural networks to ALEKS data [21, 22, 24], none of these previous models took into account the specific details of the knowledge spaces. As such, the distinguishing feature of the neural network used for the initial assessment is that it applies a specially designed architecture to output probabilities consistent with the knowledge space—in particular, this architecture ensures that the probability estimates follow the prerequisite relationships among the topics. Thus, the neural network can leverage the vast amounts of ALEKS data to make accurate predictions, while simultaneously respecting the set of knowledge states and thereby ensuring these predictions are pedagogically sound. Notably, similar ideas have been successfully applied in knowledge tracing models, where it’s been shown that leveraging prerequisite relationships can be done effectively through the loss function of a neural network [2], or by utilizing dynamic Bayesian networks [19].

¹See [23] for evidence of a ‘fatigue effect’ experienced by students towards the end of an ALEKS assessment.

²The distinction between the unknown and uncertain topics is mainly relevant for the student’s learning in the system, with these categorizations determining the amount of work required before a topic is considered to be mastered.

Solve for x .

$$2(3x - 6) = 12$$

Simplify your answer as much as possible.



Figure 1: Screen capture of the ALEKS topic “Introduction to solving an equation with parentheses.”

3. EXPERIMENTAL SETUP

Neural network models were trained for eight ALEKS products: middle school math courses 1-3 (in the U.S., these correspond to grades 6 through 8); general chemistry A and B (the first and second semesters of college-level general chemistry, respectively); and three college-level math classes (pre-calculus, college algebra, and college algebra with trigonometry). Although we do not have access to specific demographic information for the students in our data, we can say that the majority of the middle school users are from U.S. public schools, while the higher education users come from both community colleges and four-year institutions, again mainly from the U.S. We began evaluating the new assessment engine in mid-December of 2021, using A/B testing to compare its performance to that of the existing version of the assessment, with data being collected through February of 2022. For the aforementioned products, we used a cluster randomized design to assign the two versions of the assessment at the class level, an approach commonly used for educational studies [14, 27, 29]. Specifically, each class was randomly assigned to receive either the new assessment or the old assessment, whereupon all the students in that class then received the same version of the assessment.

For the middle school and chemistry products, the neural network models were trained with the exact data sets that were used to build the models for the existing version of the ALEKS assessment engine; as such, our analyses on these products are the most informative when trying to precisely estimate the differences in performance between the two engines. In comparison, rather than being restricted to a certain time period, the college math neural network models were trained on all the available data. Thus, from the purely scientific perspective of measuring the differences between the assessment engines, comparing the results on the college math courses is perhaps less useful. However, from the more practical standpoint of understanding the (possible) gains from updating existing ALEKS products with the new engine, we believe this analysis to be informative. That is, there are currently many older ALEKS products that haven’t been updated over the past several years; thus, if these products were updated with the new neural network assessment engine, they would stand to gain from both the application of the neural network *and* the extra data used to train the model. As such, we believe this analysis gives some insight into the potential benefits from converting these older ALEKS products to the new assessment engine.

Table 1: Summary statistics for the three groups of ALEKS products.

Product	Type	Total		Average number of topics	Extra problem correct rate
		Classes	Assessments		
Middle school math	KST	5,311	19,480	433.6	0.334
	NN	5,407	19,313	436.4	0.330
Chemistry	KST	360	16,315	158.0	0.264
	NN	381	18,812	155.2	0.257
College math	KST	1,247	33,207	258.2	0.317
	NN	1,202	30,785	253.9	0.311

4. ANALYSIS

In order to compare the performance of the two assessment models, in what follows we make use of an *extra problem* that is asked during each initial assessment. This extra problem is chosen uniformly at random from the topics in the course and presented to the student as an assessment question. However, the answer to the extra problem does not affect the results of the assessment—instead, the information from the extra problem is used to evaluate and improve the ALEKS system. In Table 1 we show summary statistics after partitioning the products into three distinct groups—middle school math (middle school math courses 1–3), chemistry (general chemistry A and B), and college math (precalculus, college algebra, and college algebra with trigonometry)—and also the two treatment arms of our analysis—the KST-based assessment, and the neural network assessment (NN). In addition to showing the total numbers of classes and initial assessments taken, we also show the average number of topics used by each class, weighted by the number of assessments per class.³ Finally, in the last column we display the average correct answer rate to the extra problem.

Our first analysis compares the performance of the two assessment models by treating them as binary classifiers, where we consider a positive outcome to be a correct answer to the extra problem, while a negative outcome is either an incorrect or “I don’t know” response. We compare the assessment engines using three metrics: area under the receiver operating characteristic curve (AUROC), point biserial correlation (r_{pb}), and accuracy score. AUROC is frequently used to evaluate probabilistic classifiers, and it is known to perform well even with some class imbalance [16]. The point biserial correlation is a special case of the Pearson correlation coefficient in which one variable is dichotomous (the student response) and the other variable is continuous (the probability estimate from the assessment).⁴ While the actual probabilities are used in the computations of AUROC and the point biserial correlation, for the accuracy calculation we

assume any probability at or above 0.5 is a positive prediction, with anything below 0.5 then being considered a negative prediction. We should clarify that while this assignment of prediction labels is a standard procedure used to evaluate binary classifiers, it does not necessarily correspond to the actual classifications made by the ALEKS system—we look at these ALEKS-specific classifications in more detail later.

As students are grouped—or, “clustered”—into classes, to compute confidence intervals around the point estimates we use the following bootstrap procedure, applied separately to each product and assessment engine pair. That is, we apply this procedure a total of six times: once for the middle school products using the KST engine, then for the middle school products using the neural network engine, then for the chemistry products with the KST engine, etc. The first step in the procedure is to resample our data using the *cluster bootstrap*, a modified version of the standard bootstrap that specifically works with clustered data [17]. For our analysis, classes are randomly sampled with replacement from our original data set, until we have a sample of classes equal to the number in our original data set. Then, we combine all the assessments from these selected classes to generate one bootstrap sample—note this means that some classes, as well as the associated assessments, appear multiple times in the sample. Next, we compute our statistics for this bootstrap sample, and we then repeat this entire procedure until we’ve generated 20,000 bootstrap samples in total. Finally, since each resulting bootstrap distribution turns out to be symmetric and centered at the original values of the statistic—i.e., the value of the statistic computed from our original set of data—we compute each confidence interval by simply taking the 2.5th and 97.5th percentiles.

The results are shown in Table 2. For the middle school products, we can see that the neural network engine performs better according to each of the metrics, even after taking into account the confidence intervals. Next, the results for the chemistry products are less conclusive. Although the point estimates are all higher for the neural network engine, based on the confidence intervals in the third column there’s uncertainty with the signs of these differences. Finally, for the college math products the neural network engine again has much stronger performance compared to the KST-based assessment—we reiterate that this is expected, as the college math neural network models had access to more recent data in comparison to the corresponding KST models.

While the results for the middle school and college math

³While each course has a default set of recommended topics, the instructor is free to add or remove topics from this set.

⁴The Matthews correlation coefficient (MCC) [25] is a related statistic that is regarded as being an informative measure for evaluating binary classifiers [1, 3, 4, 26]. As the MCC is mathematically equivalent to the Pearson correlation coefficient of two dichotomous variables—also known as the phi coefficient—the only difference from computing the point biserial correlation is that the MCC requires we dichotomize the probability estimates. However, since this would result in some loss of information, we prefer to use the point biserial correlation for our current evaluations.

Table 2: Comparison of the two assessment engines, using the area under the receiver operating characteristic curve (AUROC), point biserial correlation coefficient (r_{pb}), and accuracy score. Numbers in parentheses represent the 95% confidence intervals computed from 20,000 cluster bootstrap samples.

Product	Metric	Assessment type		Difference
		KST	NN	NN–KST
Middle school math	AUROC	0.875 (0.870, 0.881)	0.894 (0.889, 0.899)	0.019 (0.011, 0.026)
	r_{pb}	0.624 (0.612, 0.635)	0.674 (0.664, 0.684)	0.050 (0.035, 0.065)
	Accuracy	0.811 (0.805, 0.817)	0.831 (0.825, 0.836)	0.020 (0.012, 0.028)
Chemistry	AUROC	0.871 (0.861, 0.881)	0.880 (0.873, 0.888)	0.009 (-0.003, 0.022)
	r_{pb}	0.610 (0.585, 0.634)	0.629 (0.613, 0.646)	0.019 (-0.010, 0.049)
	Accuracy	0.837 (0.829, 0.845)	0.838 (0.827, 0.849)	0.001 (-0.013, 0.014)
College math	AUROC	0.861 (0.856, 0.867)	0.898 (0.893, 0.902)	0.036 (0.029, 0.043)
	r_{pb}	0.599 (0.589, 0.610)	0.674 (0.664, 0.684)	0.075 (0.060, 0.089)
	Accuracy	0.814 (0.807, 0.821)	0.838 (0.833, 0.843)	0.024 (0.015, 0.032)

products are encouraging, the performance of the chemistry neural network models is slightly unexpected. Based on our previous evaluations when training the neural network models, we expected a larger performance improvement over the KST-based assessment for the chemistry products—in comparison, the performance of the middle school and college math neural network assessments are consistent with our expectations. A possible concern is that there are differences between the chemistry student populations using the two different assessment engines—while not conclusive, some of the statistics in Table 1 are suggestive of such a difference. To start, although there are over 35,000 total assessments taken for the chemistry products, the number of unique classes is low in comparison to the other products—for example, the number of chemistry classes (741) is a small fraction of the number of middle school classes (10,718). This is important, since in a cluster randomized experiment such as ours, the number of clusters is typically more restrictive than the overall sample size [29], and such designs have a higher risk of non-equivalence between the experimental groups [14, 27]. Additionally, for the chemistry products the neural network group has about 15% more assessments taken than the KST group, which is possibly another sign of non-equivalence between the groups. While these differences aren’t conclusive, they at least suggest that the student populations may differ in some respect. Thus, in the next section we analyze the chemistry products further in order to obtain a better understanding of the results.

5. REANALYZING THE CHEMISTRY ASSESSMENTS

To investigate the possibility that the student populations are not equalized across the chemistry experimental groups, we take advantage of the fact that an ALEKS assessment can be “replayed” on an assessment engine different from the one that was originally used. For example, suppose a student takes an assessment using the KST-based engine. Once this assessment is completed, we can feed the questions and responses to the neural network engine—taking care to remove the extra problem from this process—generate probability estimates, and then evaluate the probability for the extra problem in the original assessment. The main

drawback to this approach is that the engine used for the replay assessment won’t be able to choose the questions that are given to the student, which could theoretically bias the results somewhat. However, the advantage of this approach, and the reason we employ it here, is that it allows us to directly compare the assessment engines on the same sets of data, removing any concerns about the non-equivalence of the experimental groups.

To that end, using the data from the 16,315 chemistry assessments originally taken with the KST engine, we feed the questions and responses to the neural network models and generate probability estimates. We then take these probabilities and compute our evaluation metrics on the extra problems. Next, we repeat the same procedure in the other direction—that is, using the data from the 18,812 chemistry assessments that originally used the neural network engine, we take the questions and responses from each assessment and feed them to the KST-based engine. As before, we use the resulting probabilities to compute our evaluation metrics on the extra problems.

Table 3: Comparison of the replayed assessments on the chemistry products. Numbers in parentheses represent the 95% confidence intervals computed from 20,000 cluster bootstrap samples.

Metric	Assessment type		Difference
	KST	NN	NN - KST
AUROC	0.856	0.889	0.033
	(0.847, 0.866)	(0.881, 0.898)	(0.020, 0.046)
r_{pb}	0.581	0.651	0.070
	(0.561, 0.602)	(0.630, 0.671)	(0.041, 0.099)
Accuracy	0.824	0.844	0.021
	(0.812, 0.835)	(0.836, 0.853)	(0.007, 0.035)

The results are shown in Table 3, where we can see a large difference in performance between the two assessment engines. In contrast to the results from Table 2, on the replayed assessments the neural network engine does much better, while the performance of the KST assessment engine has dropped noticeably. As such, the contrast in performance between the assessment engines is clearer, with

the 95% confidence intervals for the differences (third column) all bounded away from zero. So, it does appear likely that there are underlying population differences between the groups of students using the two assessment engines. Thus, arguably the fairest comparison is to use both the actual assessment data and the replayed assessment data, and then compare the assessment engines based on this combined data set. The results are shown in Table 4, where we can see a relatively clear performance gap between the two engines, albeit not quite as large as in Table 3.

Table 4: Combined comparison of the two assessment engines on the chemistry products, including data from both the original and replayed assessments. Numbers in parentheses represent the 95% confidence intervals computed from 20,000 cluster bootstrap samples.

Metric	Assessment type		Difference NN - KST
	KST	NN	
AUROC	0.863 (0.856, 0.870)	0.885 (0.879, 0.891)	0.022 (0.012, 0.031)
r_{pb}	0.594 (0.578, 0.610)	0.639 (0.626, 0.653)	0.045 (0.024, 0.067)
Accuracy	0.830 (0.823, 0.837)	0.841 (0.834, 0.848)	0.011 (0.001, 0.021)

6. ALEKS KNOWLEDGE STATES

While the previous analyses have compared the performance of the assessment engines assuming they are standard binary classifiers, in this section we compare the models based on measures more specific to the ALEKS system. In what follows, we restrict our analyses to the middle school data, as we believe this data set gives the most balanced and fair comparison between the two assessment engines.

Recall that the purpose of the initial assessment is to identify the topics in a student’s knowledge state—that is, the topics in the known category. Since the ALEKS system uses this information to determine what a student is ready to learn, inaccurately measuring a knowledge state would negatively affect the student’s learning experience. For example, giving a student credit for many topics that aren’t justified could cause the student to start their learning with topics for which they aren’t prepared, possibly leading to frustration. On the other hand, not giving a student enough credit for topics they know has an opposite effect, as the student may start with topics that are too easy, causing boredom.

To start, in Figure 2 we show a relative frequency histogram of the number of topics classified as known after each assessment is completed. The striped (blue) bars represent the 19,313 assessments from the neural network engine, while the solid bars show the proportions for the 19,480 assessments from the KST engine. The mean and median are 112.5 and 98, respectively, for the neural network engine; in comparison, these values are 106.9 and 85 for the KST engine. Depending on whether we use the mean or median to describe the outcome of a “typical” assessment, the knowledge states from the neural network assessment engine are larger by either 5.2% (mean) or 15.3% (median). Furthermore, the first and third quartiles for the neural network

engine are 52 and 159, respectively, with these values being 42 and 152 for the KST engine. These differences possibly indicate that the advantages of the neural network engine apply to a diverse sample of students, rather than only those in a specific part of the distribution. Moreover, it’s encouraging that the gains are larger for the students in the first quartile, as the relative benefit of the additional topics is greater for students with smaller knowledge states.

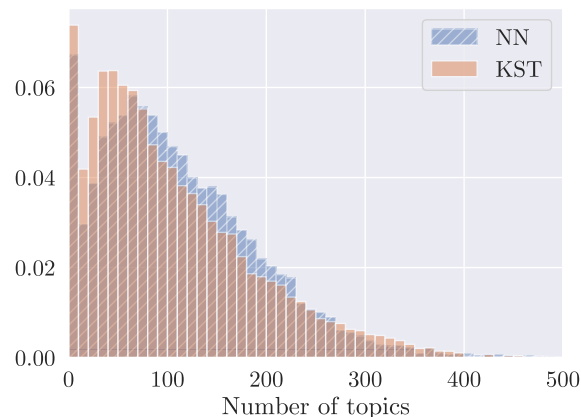


Figure 2: Relative frequency histogram of the number of known topics for the middle school products.

While the knowledge states from the neural network engine tend to be larger, it’s important that they are also accurate; assigning more topics to the known category is of limited use if the topics aren’t actually known by the students. As such, in our next analysis we look at how often students answer correctly to the extra problem based on the classification (or, categorization) of the ALEKS system—either known, unknown, or uncertain. The results are shown in Table 5. Starting with the known category, we can see that students answer correctly more often with the neural network engine compared to the KST engine—0.792 vs. 0.788. Additionally, the neural network classifies the extra problem as being known more often than the KST model—0.271 vs. 0.259—which is consistent with the results in Figure 2. Next, note that while we want a high rate of correct answers to the topics in the known category, for topics in the unknown category we want the opposite—that is, a low correct answer rate to the unknown topics indicates the classifications are accurate. We can see that the unknown topics for the neural network engine have a lower correct answer rate in comparison to those from the KST engine—at the same time, the proportions of topics labeled as unknown are comparable. Finally, the neural network engine has a lower proportion of topics in the uncertain category, showing that overall it’s more aggressive in labeling topics as known or unknown.

In this last analysis we’d like to get a different perspective on the performance of the models. In particular, both assessment engines rely on the same underlying knowledge spaces, which means the prerequisite relationships between the topics are the same. Furthermore, when a topic is answered correctly during an assessment, the ALEKS system uses these prerequisite relationships to classify topics as being known—specifically, if a topic is answered correctly, that topic, as well as all of its prerequisites, are classified as known. Since

Table 5: Statistics for the assessment engines, partitioned by the classification of the extra problem.

Classification		Assessment type	
		KST	NN
Known	Proportion	0.259	0.271
	Correct rate	0.788	0.792
Unknown	Proportion	0.573	0.574
	Correct rate	0.106	0.088
Uncertain	Proportion	0.168	0.154
	Correct rate	0.412	0.421

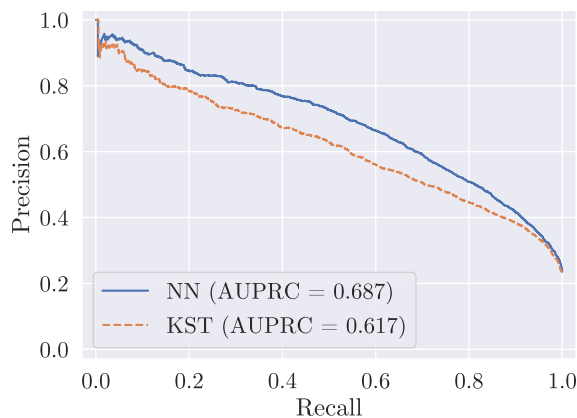


Figure 3: Precision-recall curves for the extra problems that are not directly classified by the prerequisite relationships.

this behavior is the same for each assessment engine, we want to remove these classifications and obtain a more nuanced view of the differences in performance. Additionally, these “prerequisite” classifications tend to be the easiest ones for the assessment engines to make, and so by removing them we can evaluate the engines on a more challenging subset of the data. To that end, we remove any extra problem that (a) also appeared as a regular assessment question and was answered correctly, or (b) is a prerequisite of a topic that was correctly answered during the assessment—this leaves us with 16,122 and 15,963 extra problems for the KST and neural network engines, respectively. Since the performance of the ALEKS system depends on the assessment’s ability to correctly classify the topics in a student’s knowledge state (i.e., precision), while simultaneously identifying as many such topics as possible (i.e., recall), for these data points we plot the precision-recall curves, as we can then compare the performance of the assessment engines across a range of thresholds. The results are shown in Figure 3, where we can see that the neural network curve dominates for the vast majority of the recall values, with the precision values being substantially higher in many places. Overall, it’s informative to see the strong performance of the neural network assessment engine on this specific subset of the data.

7. DISCUSSION

In this work we presented and analyzed the results of a randomized experiment—or, A/B test—comparing two differ-

ent versions of the ALEKS initial assessment engine. The purpose of this analysis was to validate the neural network assessment engine and verify it improves upon the existing version that’s based on knowledge space theory (KST). While an initial analysis indicated strong improvement for the middle school and college math products, the difference was less clear for the chemistry products. To investigate further, we reanalyzed the chemistry assessments by “replaying” each of them with the competing engine. This analysis suggested that the student populations in the two experimental groups for the chemistry products were not completely equivalent, confounding the comparison. Thus, we adjusted for the non-equivalence of the student populations by combining the data from both the replayed and actual assessments, with the results indicating that the neural network assessment outperformed the KST-based version. Finally, we evaluated the assessment engines on metrics more specific to the ALEKS system—in these analyses, we saw that the neural network assessment gave students credit for knowing more topics, while simultaneously being more accurate.

The chemistry results were interesting and somewhat surprising. Given the large numbers of students in the two experimental groups, it was unexpected that, as suggested by the replay results, these groups would be dissimilar. However, since many of the chemistry students are from large universities, the class sizes also tend to be large—thus, as seen in Table 1 the number of distinct classes is relatively small. (In comparison, the middle school products have a higher number of “independent” users—e.g., homeschooling/home education students, or individual students seeking extra help—resulting in smaller average class sizes.) In experimental designs with multilevel structure such as ours, the number of clusters—represented by the student classes in our data—is typically more important than the overall sample size in ensuring the baseline equivalence of the experimental groups [14, 27, 29]. As such, our experience highlights the fact that the use of cluster randomized designs, while desirable in education research for several reasons, can lead to difficulties with the statistical analysis of the results, and that this can be an issue even with seemingly large samples of data.

Overall, we found the performance of the neural network assessment engine to be promising, in that it has the potential to benefit students in multiple ways. For example, the fact that it returns larger knowledge states in comparison to the KST engine—and, importantly, without a drop in accuracy—means that students do not have to spend as much time working on topics they may already know. Thus, students can learn more efficiently and spend more time working on completely new material, hopefully allowing them to progress further in the course. Yet another possible benefit pertains directly to the initial assessment itself. User feedback from both students and teachers has informed us that there is a desire for a shorter initial assessment, one that asks fewer questions and takes less time to finish. Given the gain in performance, it seems plausible that the neural network assessment could still improve on the KST-based assessment even if fewer questions are asked. We are currently looking at this possibility in detail, with the hope of shortening the initial assessment and improving the student experience within the ALEKS system.

8. REFERENCES

- [1] S. Boughorbel, F. Jarray, and M. El-Anbari. Optimal classifier for imbalanced data using Matthews correlation coefficient metric. *PLoS one*, 12(6):e0177678, 2017.
- [2] P. Chen, Y. Lu, V. W. Zheng, and Y. Pian. Prerequisite-driven deep knowledge tracing. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 39–48. IEEE, 2018.
- [3] D. Chicco. Ten quick tips for machine learning in computational biology. *BioData mining*, 10(1):1–17, 2017.
- [4] D. Chicco and G. Jurman. The advantages of the matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. *BMC genomics*, 21(1):1–13, 2020.
- [5] E. Cosyn, C. Doble, J.-C. Falmagne, A. Lenoble, N. Thiéry, and H. Uzun. Assessing mathematical knowledge in a learning space. In J.-C. Falmagne, D. Albert, C. Doble, D. Eppstein, and X. Hu, editors, *Knowledge Spaces: Applications in Education*, pages 27–50. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.
- [6] E. Cosyn, H. Uzun, C. Doble, and J. Matayoshi. A practical perspective on knowledge space theory: ALEKS and its data. *Journal of Mathematical Psychology*, 101:102512, 2021.
- [7] D. de Chiusole, L. Stefanutti, P. Anselmi, and E. Robusto. Stat-Knowlab. Assessment and learning of statistics with competence-based knowledge space theory. *International Journal of Artificial Intelligence in Education*, 30(4):668–700, 2020.
- [8] M. C. Desmarais and M. Gagnon. Bayesian student models based on item to item knowledge structures. In *European Conference on Technology Enhanced Learning*, pages 111–124. Springer, 2006.
- [9] M. C. Desmarais, A. Maluf, and J. Liu. User-expertise modeling with empirically derived probabilistic implication networks. *User modeling and user-adapted interaction*, 5(3):283–315, 1995.
- [10] C. Doble, J. Matayoshi, E. Cosyn, H. Uzun, and A. Karami. A data-based simulation study of reliability for an adaptive assessment based on knowledge space theory. *International Journal of Artificial Intelligence in Education*, 29(2):258–282, 2019.
- [11] J.-P. Doignon and J. Falmagne. Knowledge spaces and learning spaces. *arXiv preprint arXiv:1511.06757*, pages 1–51, 2015.
- [12] J.-P. Doignon and J.-C. Falmagne. Spaces for the assessment of knowledge. *International Journal of Man-Machine Studies*, 23:175–196, 1985.
- [13] J.-P. Doignon, J.-C. Falmagne, and E. Cosyn. Learning spaces: A mathematical compendium. In J.-C. Falmagne, D. Albert, C. Doble, D. Eppstein, and X. Hu, editors, *Knowledge Spaces: Applications in Education*, pages 131–145. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.
- [14] J. Dreyhaupt, B. Mayer, O. Keis, W. Öchsner, and R. Muehe. Cluster-randomized studies in educational research: principles and methodological aspects. *GMS Journal for Medical Education*, 34(2), 2017.
- [15] J.-C. Falmagne and J.-P. Doignon. *Learning Spaces*. Springer-Verlag, Heidelberg, 2011.
- [16] T. Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006.
- [17] C. A. Field and A. H. Welsh. Bootstrapping clustered data. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 69(3):369–390, 2007.
- [18] C. Hockemeyer, T. Held, and D. Albert. RATH-A relational adaptive tutoring hypertext WWW-environment based on knowledge space theory. In *Proceedings of 4th International Conference on Computer Aided Learning and Instruction in Science and Engineering*, pages 417–423. Citeseer, 1997.
- [19] T. Käser, S. Klingler, A. G. Schwing, and M. Gross. Beyond knowledge tracing: Modeling skill topologies with bayesian networks. In *International conference on intelligent tutoring systems*, pages 188–198. Springer, 2014.
- [20] D. Lynch and C. P. Howlin. Real world usage of an adaptive testing algorithm to uncover latent knowledge. In *Proceedings of the 7th International Conference of Education, Research and Innovation*, pages 504–511, 2014.
- [21] J. Matayoshi and E. Cosyn. Identifying student learning patterns with semi-supervised machine learning models. In *Proceedings of the 26th International Conference on Computers in Education*, pages 11–20, 2018.
- [22] J. Matayoshi, E. Cosyn, and H. Uzun. Are we there yet? Evaluating the effectiveness of a recurrent neural network-based stopping algorithm for an adaptive assessment. *International Journal of Artificial Intelligence in Education*, 31(2):304–336, 2021.
- [23] J. Matayoshi, U. Granzio, C. Doble, H. Uzun, and E. Cosyn. Forgetting curves and testing effect in an adaptive learning and assessment system. In *Proceedings of the 11th International Conference on Educational Data Mining*, pages 607–612, 2018.
- [24] J. Matayoshi, H. Uzun, and E. Cosyn. Deep (un)learning: Using neural networks to model retention and forgetting in an adaptive learning system. In *International Conference on Artificial Intelligence in Education*, pages 258–269. Springer, 2019.
- [25] B. W. Matthews. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, 405(2):442–451, 1975.
- [26] D. M. Powers. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. *Journal of Machine Learning Technologies*, 2:37–63, 2011.
- [27] S. W. Raudenbush. Statistical analysis and optimal design for cluster randomized trials. *Psychological Methods*, 2(2):173–185, 1997.
- [28] A. Reddy and M. Harper. Mathematics placement at the University of Illinois. *PRIMUS*, 23(8):683–702, 2013.
- [29] T. A. Snijders and R. J. Bosker. *Multilevel Analysis: An Introduction to Basic and Advanced Multilevel Modeling*. Sage, 2011.

Mining Artificially Generated Data to Estimate Competency

Robby Robson¹, Fritz Ray², Mike Hernandez³, Shelly Blake-Plock⁴, Cliff Casey⁵, Will Hoyt⁶, Kevin Owens⁷, Michael Hoffman⁸, Benjamin Goldberg⁹

ABSTRACT

The context for this paper is the *Synthetic Training Environment Experiential Learning – Readiness* (STEEL-R) project [1], which aims to estimate individual and team competence using data collected from synthetic, semi-synthetic, and live scenario-based training exercises. In STEEL-R, the *Generalized Intelligent Framework for Tutoring* (GIFT) orchestrates scenario sessions and reports data as experience API (xAPI) statements. These statements are translated into assertions about individual and team competencies by the *Competency and Skills System* (CaSS). Mathematical models use these assertions to estimate the competency states of trainees. This information is displayed in a dashboard that enables users to explore progression over time and informs decisions concerning advancement to the next training phase and which skills to address.

To test, tune, and demo STEEL-R, more data was needed than was available from real-world training exercises. Since the raw data used to estimate competencies are captured in xAPI statements, a component called DATASIM was added. DATASIM simulated training sessions by generating xAPI statements that conformed to a STEEL-R *xAPI Profile*. This facilitated testing of STEEL-R and was used to create a demo that highlighted the ability to map data from multiple training systems to a single competency framework and to generate a display that team leaders can use to personalize and optimize training across multiple training modalities.

This paper gives an overview of STEEL-R, its architecture, and the features that enabled the use of artificial data. The paper explains how xAPI statements are converted to assertions and how these are used to estimate trainee competency. This is followed by a section on xAPI Profiles and on the xAPI Profile used in STEEL-R. The paper then discusses how artificial data were generated and the challenges of modeling longitudinal development and team in these data. The paper ends with a section on future research.

1. INTRODUCTION

The research reported in this paper relates to the US Army *Synthetic Training Environment* (STE) initiative that “brings together live and virtual training environments, aiming to deliver accessible exercises that mimic the full complexity of the physical world” [2]. To support this initiative, the initiative is developing infrastructure and a suite of *Training Management Tools* (TMT) that permit diverse training systems – including desktop game-based, mixed reality, virtual reality, augmented reality, and sensor-instrumented live training – to be rendered and integrated in a single training environment and training to be optimized within this environment.

R. Robson, B. Goldberg, S. Blake-Plock, C. Casey, W. Hoyt, M. Hernandez, and F. Ray. Mining artificially generated data to estimate competency. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 828–833, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852926>

The goal of the *STE Experiential Learning – Readiness* (STEEL-R) project is to support the STE TMT with software that collects evidence from training scenarios and uses this to estimate team and individual competency and performance probabilities, recommend training interventions, and inform the design of training scenarios. Research to date has focused on US Army battle drills [3], i.e., short tactical team scenarios intended to train individuals and teams to an automated response level. These include cognitive, psychomotor, and affective skills and behaviors that can be trained in a series of training systems that progress from first-person shooter game-like environments such as Virtual Battle Space 3 [4] (*synthetic* in this paper), to mixed reality and augmented reality environments (*semi-synthetic*), and field exercises in which trainees are instrumented with sensors (*live*). Traditionally, observer controllers / trainers (OC/Ts) are present and can alter conditions on the fly to change difficulty or add stressors. A goal of the STE TMT is to accelerate development and skill retention by using data-driven automation and the capabilities of intelligent tutoring systems to support assessment and facilitate personalized coaching.

The training addressed by STEEL-R is *experiential*, meaning that learning and mastery require repeated deliberate practice under varied conditions. To support experiential skill acquisition, the underlying competency and predictive models in STEEL-R must take longitudinal data and progression into account. This type of training also heavily involves team tasks and *team dimensions* such as cohesion, communication, and backup behaviors [5]. This adds further complexity to the underlying models and places further requirements on the data that must be collected to generate and test these models. Since the demand for such data is too large to be met by small trials, and since it is important that STEEL-R demonstrate good results prior to deployment in high-stakes real-world training, we saw *artificial data* as the best way to proceed in the early and middle stages of our research. We use the term *artificial* rather than *synthetic* in this paper to avoid confusion with synthetic training.

This paper focuses on the use of artificial data, on the use of xAPI Profiles and DATASIM to produce artificial data, on the challenges encountered, and on the results obtained. We start with an overview of the STEEL-R architecture and its critical features, which is next.

2. STEEL-R ARCHITECTURE

Three systems play a central role in the STEEL-R architecture, shown in green in Figure 1. The first is the *Generalized Intelligent Framework for Tutoring* (GIFT) [6], which orchestrates scenario sessions. It connects to and collects data on trainee actions from training systems via connectors. GIFT examines these actions and assesses whether specified tasks, activities, and expected behaviors are performed or demonstrated at, below, or above expectations.

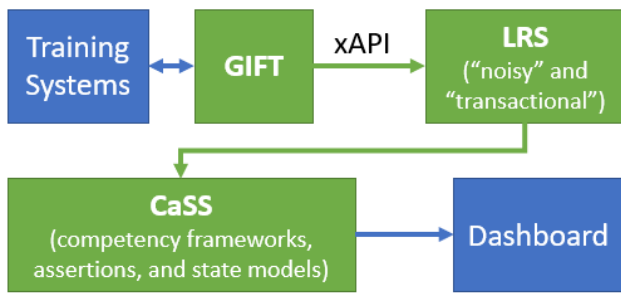


Figure 1: STEEL-R Component Architecture

The second core component is a set of two *Learning Record Stores* (LRSs). Every time an assessed performance state changes, GIFT reports the state and associated session data to a “noisy” LRS via experience API (xAPI) [7] statements. This noisy LRS captures everything that is happening in a session. A second “transactional” LRS filters noisy LRS data, retaining only the overall assessment of each trainee in a session. Data from the transactional LRS is polled by an instance of the *Competency and Skills System* (CaSS) [8], the third core component.

CaSS stores *competency objects* that represent the individual and team tasks, skills, knowledge, attitudes, and behaviors that a given instance of STEEL-R is intended to train and track. These are stored in *competency frameworks* that include relations among the skills, competencies, and behaviors these objects represent. As STEEL-R runs, CaSS collects xAPI statements and formulates *assertions*. An assertion is a statement to the effect that a trainee or team has or has not demonstrated competency based on identified evidence under specified conditions [9].

A GIFT performance assessment generated can generate multiple assertions about multiple competency objects. As explained in Section 3, CaSS computes competency states from these assertions using mathematic models. Competency states and other data are sent by CaSS to a Dashboard that can be used to view states, track progress, and make informed training decisions. Throughout this process, the chain of evidence is preserved so an OC/T can review and audit it. Dashboard data can be traced back through CaSS, the LRS, and GIFT to trainee actions that can be replayed in a GIFT “gamemaster” interface.

A crucial feature of this architecture is that the data from all training systems is filtered through GIFT, where it is referenced against a common set of competency objects. Data from desktop games, mixed reality simulations, and live exercises can thus be combined to estimate and track the state of each single skill and competency.

2.1 STEEL-R Implementation

STEEL-R uses a Multiple Open System Architecture (MOSA) that integrates GIFT and the US Advanced Distributed Learning (ADL) initiative’s Total Learning Architecture (TLA) [10] [11]. Most components (including GIFT and CaSS) are customized versions of open-source software. The LRSs are instances of the Yet Analytics SQL LRS [12], and all components are hosted on a hybrid container-based platform (Docker™). Since STEEL-R is intended to support field operation, it is designed for offline use and the entire system can be deployed on a mid-range rugged laptop. In field deployments, STEEL-R collects data in offline mode and forwards it to a cloud hosted instance when it comes back online. The mechanisms that permit offline operation enable STEEL-R to assemble data received from multiple sources at different times into

a coherent sequence of events along a single timeline. This feature turned out to be crucial, as will be discussed in Section 5.2.

3. ESTIMATING COMPETENCY

The current version of STEEL-R considers three competency states – *untrained*, *practiced*, and *trained* – and three training phases – *crawl*, *walk*, and *run*. The three states are derived from US Army doctrine, while the training phases roughly correspond to synthetic, semi-synthetic, and live training. The Dashboard component of STEEL-R informs OC/Ts how an individual or team is progressing from untrained to practiced to trained within each training phase and how ready they are for the next training phase, i.e., to move from crawl to walk and walk to run. In future Army versions, an *expert* state may be added, and different states and phases may be used for different application domains.

3.1 The Math Model

CaSS populates the STEEL-R Dashboard with longitudinal data about the competency state of each individual or team. These states are estimated using a mathematical model (the “math model”) that involves a *repetition function*, an *evidence function*, and *rollup rules*. The repetition function represents the number of times a skill or competency has been trained, weighted by a forgetting function and a function that accounts for the value of spaced repetition. Similar methods are used in ACT-R [13] and the work of Jastrzembski and others on predicting future training performance [14]. The evidence function assigns a score between -1 and 1 that is derived from the history of performance assessments, taking skill decay, the trustworthiness of the evidence, and performance on related skills and competencies into account. Rollup rules allow performance on related skills and competencies to contribute to an evidence function and allow dependencies on performance under varied conditions and on the states of sub-skills to be added. More details model can be found in [15].

3.2 The Role of CaSS Assertions

The raw data used to evaluate the repetition and evidence functions comes from *assertions*. Assertions are a fundamental data type in CaSS that expresses conclusions drawn from evidence in a uniform way. An assertion can identify its source, the on which the source relied, the source of the evidence, the person or team and the CaSS competency object about which the assertion is made, a timestamp, a decay function, and a parameter that indicates the confidence its source has in the assertion [15]. Assertions can assert that a skill or competency was or was not demonstrated or is or is not possessed. Assertions can also identify the conditions under which the evidence was gathered e.g., difficulty factors and stressors.

CaSS computes the repetition and evidence functions from data in assertions. For example, if a competency object represents the ability of a team or person to perform a task, CaSS examines all assertions about their performance on that task and uses these to determine when and how often and with what results the task was attempted under varied conditions. This information is used to compute the repetition and evidence functions, which are in turn used to estimate whether they are untrained, practiced, or trained within the crawl, walk, and run phases.

3.3 Generating Assertions from xAPI

GIFT does not directly make CaSS assertions. Instead, it assesses actions, activities, and expected behaviors based on data from a training system and emits xAPI statements that require translation into CaSS assertions. CaSS does this with a *decoder*. The decoder has a lookup table that maps activities to competency objects and specifies how the three states reported by GIFT (at, above, and

below expectations) translate into positive or negative assertions about these objects. At present, this lookup table is hard-coded based on subject matter expert (SME) input.

3.4 The Need for Data

The math model and decoder have weights and parameters that can be set in a STEEL-R instance and control competency estimates. As STEEL-R develops, these will be used to compute performance probabilities and to recommend interventions and scenario designs. The system is designed so that its weights and parameters can be machine-learned, but at this stage they are manually set based on experimentation guided by theory. This requires significant data, and machine learning will require even more.

The best data would be data from real-world training exercises. Unfortunately, there are limited opportunities to deploy STEEL-R in such exercises, and since many involve high-stakes training, STEEL-R must be thoroughly stress-tested and shown to produce reasonable results before deployment can be considered. For these reasons, we took the approach of generating artificial data.

Referring to Figure 1, there is a choice as to where artificial data is inserted. One choice is to inject it into CaSS in the form of assertions. This can be used to test the math model, and early on we developed a small web app to do this, see Section 5. This allowed us to check formulas and code and to demonstrate how evidence affected competency estimates, but it was not sufficient to test the entire architecture. As a result, we decided to generate artificial xAPI statements that mimicked those generated by GIFT. This involves xAPI profiles, which are explained next.

4. XAPI PROFILES

The experience API (xAPI) is a mechanism for reporting and retrieving learner activities in an *actor – verb – object – context – results* format [7], [16]. xAPI *statements* in this format are sent to an LRS where they can be retrieved by other systems with appropriate permissions. xAPI statements are usually generated by an education or training system such as an LMS, simulation, or intelligent tutoring system, but statements can come from another LRS, as is done in the STEEL-R handoff between the noisy and transactional LRS. This ability enables multiple LRSs at the edge of a network to feed a central LRS, which improves scalability. The xAPI specification, which includes specifications for LRSs, was first developed by the ADL and is now undergoing more formal standardization in IEEE [16].

The xAPI specification is intended to be usable in any education or training ecosystem. To maintain flexibility, it does not specify the context or semantics of xAPI statements. In implementations it is necessary to add definitions and place restrictions on the format and elements in statements to ensure that data is properly reported and interpreted. This is done via *xAPI Profiles* [17].

xAPI Profiles define concepts, templates, patterns, and extensions for use in forming xAPI statements. *Concepts* define the vocabulary and attributes that may appear in xAPI statements, including verbs and activity types, and specify rules for how and when they can be used. *Templates* provide rules for constructing statements. *Patterns* are collections and sequences of templates that describe the actions associated with a task, performance, or learning path. *Extensions* enable new (externally defined) attributes to be used in statements. Together, these rules and definitions enable xAPI statements to be properly formed and interpreted. The xAPI profiles specification [18] establishes rules for serializing profiles in JavaScript Object Notation (JSON) and in JSON for Linked Data (JSON-LD). Using

JSON-LD, vocabulary can link to the same or similar terms in other profiles, creating a semantic web of xAPI statements.

4.1 Designing the STEEL-R xAPI Profile

A critical factor in designing any xAPI Profile is creating concepts, statement templates, and patterns that are flexible enough to be used in many different scenarios but restrictive enough to enable data to be reported and understood in use cases of interest. The challenge for STEEL-R is that STEEL-R is meant to support many types of experiential learning. Even in relatively narrow domains, it may be necessary to track and capture data on hundreds (or more) tasks, activities, and behaviors. Profiles could be created that specify the names of tasks and performance levels for each domain, but a more flexible approach is enabled by exploiting the capabilities of GIFT.

4.2 The STEEL-R xAPI Profile

As a scenario session unfolds, GIFT determines if performance on tasks, activities, and behaviors stored in a Domain Knowledge File (DKF) exceeds, meets, or is below expectations [19]. At present, CaSS only uses summative assessments at the session level, but GIFT generates a formative assessment each time a performance state changes and can record information about the exercise, such as the conditions under which performance was assessed.

The xAPI profile designed for STEEL-R uses statement templates that enable tasks, activities, and behaviors to be referenced from the DKF and that report results on the GIFT three-step scale. This simplifies the form of statements, leaving the list of specific activities to GIFT. The STEEL-R templates also allow scenario conditions to be reported and include extensions for linking a GIFT assessment to a recording of the session segment that produced it.

To form a complete chain of evidence, STEEL-R xAPI statements can capture current performance states, changes in performance state, the factors that changed, and the conditions present when the state changed. Of particular interest to our future research is the ability to identify stressors and difficulty factors, both of which can be dynamically altered mid-session. Stress and difficulty are now being included in xAPI statements as discrete variables that are evaluated by GIFT and that CaSS can use in its math models.

Patterns represent the lifecycle of trainee participation in a training session. The templates in these patterns are populated by system events such as starting or joining a session, interactions within the session that could result in changes to a trainee's psychomotor, cognitive, or affective state, and GIFT's conclusions about a learner's overall performance with respect to specified tasks. Event data reported by GIFT is used to select the appropriate template and to filled in the template based template rules.

5. GENERATING ARTIFICIAL DATA

For testing purposes, artificial data was generated in two ways. The first was through a small app that was purpose-built to test and demonstrate the math model. This app allows users to apply hard-coded assertions about competency objects in a framework and displays how the repetition and evidence functions, competency state, and performance probabilities change with each statement.

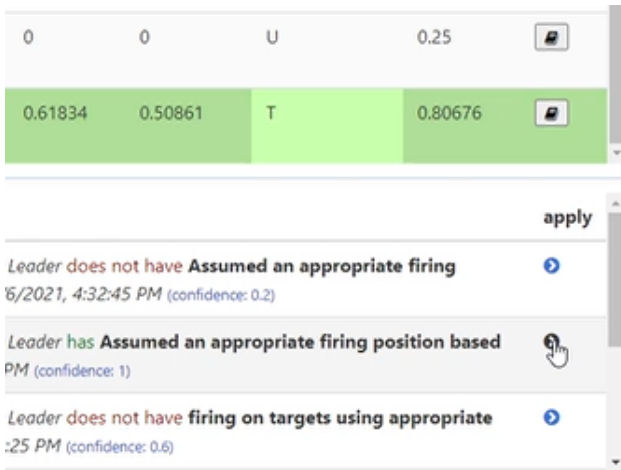


Figure 2: User applying a competency assertion

A second screen shows how the estimated probability of successful performance varies over time as both positive and negative assertions are activated. This visualization proved useful for both demonstrating and validating the math model.

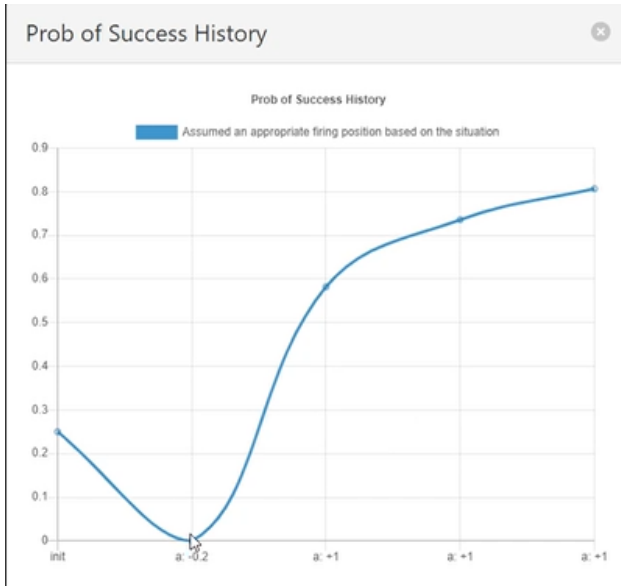


Figure 3: Display showing how applying assertions changed the estimate of performance probability as evidence accumulated.

5.1 DATASIM

The second method used to generate artificial data used an open-source component of the ADL's TLA reference implementation [11] called the *Data and Training Analytics Simulated Input Modeler* (DATASIM) [20]. DATASIM can produce xAPI datasets that conform to one or more xAPI Profiles at small scale (tens to thousands of statements) and large scale (over a billion statements). DATASIM is controlled by a *simulation specification* that a user defines via a user interface. Each simulation specification includes an xAPI Profile, the actors in the simulation, and parameters that specify the involvement of each actor in each type of activity as well as start and stop times, a seed value, and the maximum number of xAPI statements to be generated.

Within a given simulation, DATASIM generates a pseudo-random autoregressive moving average (ARMA) time series (called the

common time series) and a pseudo-random ARMA time series for each actor, each determined by the seed value. An actor generates an xAPI statement whenever their time series graph crosses the common time series graph [21]. When that happens, a Gaussian that is weighted by parameters in the simulation profile is sampled for each possible pattern and the pattern with the highest value is used. The same is then done to select templates in the pattern, statements in each template, and concepts in each statement.

5.2 Applications to STEEL-R

We used DATASIM to benchmark and stress-test STEEL-R, which was the original purpose of DATASIM [21]. By using the same actors, we could simulate multiple successive training sessions across each training phase and by referencing the same competency objects in the CaSS decoder, we could generate assertions about same competencies, skills, and behaviors at each stage. This let us validate system operation and benchmark performance at every point and for every component downstream of GIFT. It did not, however, give us the desired level of realism.

Although xAPI profiles enable DATASIM to generate data that statistically reflects the right mix of training events and outcomes at the macro level, DATASIM has no mechanism that allows a given actor to develop competency as they engage in successive activities and no mechanism to realistically correlate individual and team behavior. Thus, if DATASIM is used to simulate people performing a series of tasks, it will produce about the right number of successful and unsuccessful task completions but cannot alter its parameters during a simulation so that a person who successfully completes the early tasks will be more likely to successfully complete later ones. Similarly, if one of the actors is a team, DATASIM, the probability of team events cannot be changed based on the activities of team members during a simulation. Since longitudinal data and developmental progression are fundamental to experiential learning, we needed a way to reflect individual development and team dynamics.

We did this by running series of micro-simulations instead of one large one and by manually set parameters between runs. Each micro-simulation produced data for the same actors in a small time slice. This gave us greater control over the progression of outputs and implied team dynamics. STEEL-R treated these as offline data, automatically stringing them together to create a sequence of activities along a single timeline. This produced enough data to test and tune STEEL-R and to implement the use case described next.

5.3 A Use Case and Implementation

In November of 2021 we used DATASIM to implement and demo a use case in which a small team underwent three days of training. Our goal was to highlight how team competency improved and progressed from crawl to walk as interactions with multiple training systems activated cognitive, psychomotor, and affective skills.

On day one, the team trained on Army battle drill 6 (BD6) [3] in a synthetic game-based environment. This was done in multiple sessions with under varied conditions and with varied difficulty levels. DATASIM micro-simulations were manually configured to show performance improvement over the course of day one. On day two, BD6 training continued in a mixed reality environment that activated psychomotor skills. The data from day one showed that the team knew what to do, so day two provided opportunities to apply that knowledge in a safe controlled environment with more realistic interactions. The data generated by DATASIM represented exposure to numerous scenarios and showed further performance improvement. At the start of day three the team leader looked at the

Dashboard (Figure 4) and noticed that the team was progressing well on BD6 but there were potential skill decay issues with a related task trained in a previous battle drill. As a result, the team leader initiated training of this previous battle drill. The third day of training activated some of the same skills as the first two days and resulted in improvements in skills that seemed to have decayed.

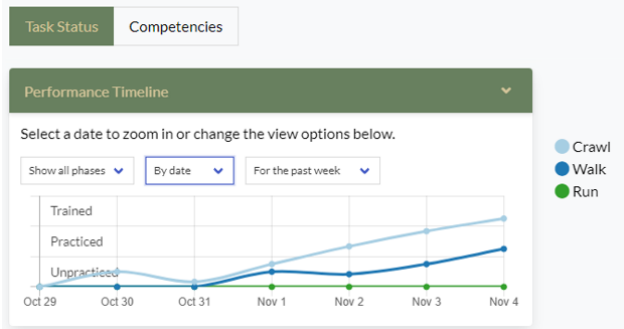


Figure 4: Part of the Dashboard, showing progress on reported by CaSS and derived from artificial data.

6. CONCLUSION

The use case we implemented and demonstrated with artificial data showed the art of the possible. It showed a team leader using competency estimates derived from multiple and varied training environments to personalize a training plan and the potential to optimize training time by leveraging multiple training modalities within a given training cycle. The ability to mine artificial data enabled us to stress-test and benchmark STEEL-R and permitted us to visualize the effects of parameters in the formulas and models used to estimate competency. This served as an excellent tool for debugging, tuning, and demonstrating the models, and we continue to take this approach as we make changes and add features.

We note that as of November of 2021, neither difficulty or stress levels were reported by GIFT or used by CaSS. These are critical factors that can be manipulated during training sessions and that should be used when determining whether an individual or team is trained and ready to advance. GIFT is now reporting difficulty and stress levels in xAPI statements, and we are incorporating difficulty and stress into the decoder, math model, and Dashboard.

Finally, our work with DATASIM exposed the need to model the progress of simulated individuals and to correlate individual and team behaviors. We did this with manually manipulated micro-simulations, which is labor intensive and will not scale. In this regard, we are exploring two further research directions. The first is to implement ways to dynamically alter the parameters used to generate artificial data during a single simulation. The second is to enable the parameters that control these alterations to be machine-learned. These will create a virtuous cycle wherein artificial data are used to test and tune new features and models, these features and models are used to improve real-world training, and real-world training data are used to improve the generation of artificial data.

7. ACKNOWLEDGEMENTS

This work was supported by U.S. Army Research Laboratory contract #W912CG20C0020. It is dedicated to Tom Buskirk who contributed to this project as a software developer and who died suddenly in 2021 at the age of 47. He was a joy to work with, was a talented developer, and is dearly missed by the STEEL-R team.

¹ Eduworks Corporation. robbey.robson@eduworks.com

² Eduworks Corporation. fritz.ray@eduworks.com

³ Eduworks Corporation. mike.hernandez@eduworks.com

⁴ Yet Analytics. shelly@yetanalytics.com

⁵ Yet Analytics. cliff@yetanalytics.com

⁶ Yet Analytics. will@yetanalytics.com

⁷ University of Texas at Arlington. kowens@arlut.utexas.edu

⁸ Dignitas Technologies. mhoffman@dignitastechnologies.com

⁹ U.S. Army DEVCOM Soldier Center

benjamin.s.goldberg.civ@army.mil

8. REFERENCES

- [1] B. Goldberg *et al.*, “Forging Competency and Proficiency through the Synthetic Training Environment with an Experiential Learning for Readiness Strategy,” presented at the Interservice/Industry Training, Simulation, and Education Conference (IITSEC), Orlando, FL, 2021.
- [2] A. Stone, “US Army makes headway on Synthetic Training Environment,” *Defense News*, 30-Sep-2021. [Online]. Available: <https://www.defensenews.com/training-sim/2021/09/30/us-army-makes-headway-on-synthetic-training-environment/>. [Accessed: 13-Feb-2022].
- [3] U. S. Army, “Appendix J - selected battle drills,” *Army Training Publication (ATP) 3-21.8*. 2022.
- [4] BISIMS, “VBS3,” *Bohemia Interactive Simulations*, 2022. [Online]. Available: <https://bisimulations.com/products/vbs3>. [Accessed: 04-Mar-2022].
- [5] R. A. Sottolare, C. Shawn Burke, E. Salas, A. M. Sinatra, J. H. Johnston, and S. B. Gilbert, “Designing Adaptive Instruction for Teams: a Meta-Analysis,” *International Journal of Artificial Intelligence in Education*, vol. 28, no. 2, pp. 225–264, Jun. 2018.
- [6] R. A. Sottolare, K. W. Brawner, A. M. Sinatra, and J. H. Johnston, “An updated concept for a Generalized Intelligent Framework for Tutoring (GIFT).” 2017.
- [7] ADL Initiative, “Experience API (xAPI) standard,” *Experience API (xAPI) Standard*. [Online]. Available: <https://adlnet.gov/projects/xapi/>. [Accessed: 20-Feb-2022].
- [8] ADL, “Competency & Skills System (CaSS),” *Advanced Distributed Learning Initiative*. [Online]. Available: <https://adlnet.gov/projects/cass/>. [Accessed: 04-Apr-2020].
- [9] R. Robson and J. Poltrack, “Using competencies to map performance across multiple activities,” in *Proceedings of the IITSEC*, 2017.
- [10] P. S. Gallagher, J. T. Folsom-Kovarik, S. Schatz, A. Barr, and S. Turkaly, “Total Learning Architecture development: A design-based research approach,” in *Proceedings of the IITSEC*, 2017.
- [11] ADL, “Understanding the TLA reference implementation,” *Advanced Distributed Learning Initiative*, 2022. [Online]. Available: <https://adlnet.gov/guides/tda/service-definitions/TLA-Reference-Implementation.html>. [Accessed: 28-Feb-2022].
- [12] Yet Analytics, “SQL LRS,” *SQL LRS*. [Online]. Available: <https://www.sqllrs.com/>. [Accessed: 27-Feb-2022].
- [13] J. R. Anderson, M. Matessa, and C. Lebiere, “ACT-R: A Theory of Higher Level Cognition and Its Relation to Visual Attention,” *Human-Computer Interaction*, vol. 12, no. 4, pp. 439–462, Dec. 1997.

- [14] T. S. Jastrzembski, K. A. Gluck, and G. Gunzelmann, "Knowledge tracing and prediction of future trainee performance," Florida State University, 2006.
- [15] R. Robson, X. Hu, E. Robson, and A. C. Graesser, "Mathematical Models to Determine Competencies," in *Design Recommendations for Intelligent Tutoring Systems - Competency-Based Scenario Design*, vol. 9, A. M. Sinatra, A. C. Graesser, X. Hu, B. Goldberg, A. J. Hampton, and J. H. Johnston, Eds. Orlando, FL: US Army Research Lab, 2022, pp. 107–112.
- [16] IEEE LTSC, "P9274.1.1," *JavaScript Object Notation (JSON) Data Model Format and Representational State Transfer (RESTful) Web Service for Learner Experience Data Tracking and Access*. [Online]. Available: <https://standards.ieee.org/ieee/9274.1.1/7321/>. [Accessed: 20-Feb-2022].
- [17] ADL, "xAPI-profiles: A set of documents addressing the structure of and supporting services for xAPI Profiles," 2018. [Online]. Available: <https://github.com/adlnet/xapi-profiles>. [Accessed: 27-Feb-2022].
- [18] IEEE LTSC, "P9274.2.1," *Standard for JavaScript Object Notation for Linked Data (JSON-LD) for Application Profiles of Learner Experience Data*, 2022. [Online]. Available: <https://standards.ieee.org/ieee/9274.2.1/10570/>. [Accessed: 27-Feb-2022].
- [19] GIFT, "Domain knowledge file 2021-2," *GIFT Tutoring Portal*, 2021. [Online]. Available: https://www.gifttutoring.org/projects/gift/wiki/Domain_Knowledge_File_2021-2. [Accessed: 27-Feb-2022].
- [20] ADL, "Data Simulator for TLA (DATASIM)," *ADL Initiative*, 2020. [Online]. Available: <https://adlnet.gov/projects/datasim/>. [Accessed: 24-May-2020].
- [21] S. Blake-Plock, "DATASIM: Data and Training Analytics Simulated Input Modeler," Yet Analytics, Mar. 2020.

“Closing the Loop” in Educational Data Science with an Open Source Architecture for Large-Scale Field Trials

Stephen E. Fancsali
April Murphy
Steve Ritter
Carnegie Learning, Inc.
{sfancsali, amurphy, sritter}
@carnegielearning.com

ABSTRACT

Ten years after the announcement of the “rise of the super experiment” at Educational Data Mining 2012, challenges to implementing “internet scale” educational experiments often persist for educational technology providers, especially when they seek to test substantive instructional interventions. Studies that deploy and test interventions, when informed by data-driven modeling, are often described as “close the loop” studies. Studies that close the loop attempt to link improvements in statistical and machine learning models of learning to real-world learning outcomes. After first considering challenges to internet scale experiments, we review several educational data science/mining studies that close the loop between data-driven modeling and learning outcomes. Next, we describe UpGrade, an open source architecture that, when integrated with educational technologies, helps overcome challenges to large-scale field trials (or internet scale experiments) that close the loop between data-driven work and real-world learning outcomes. In addition to describing preliminary randomized experiments that have been conducted and will soon be conducted using the architecture in two educational technology platforms, we end with a “call for contributors and integrators.” UpGrade contributors and integrators will be researchers and developers who seek to drive continuous, data-driven improvements in real-world settings where learning with technology occurs.

Keywords

A/B testing, closing the loop, educational technology, large-scale field trials, experimentation, open source software.

1. INTRODUCTION

At Educational Data Mining 2012, Stamper et al. [18] described “the rise of the super experiment” and the Super Experiment Framework (SEF), which conceptualizes data-driven educational experimentation at the lab scale, school scale, and internet scale. Lab scale experiments may have sample sizes in the range of 1-100; school scale experiments range in sample size between tens of learners and thousands of learners, and internet scale experiments may have sample sizes ranging from thousands of learners to

S. Fancsali, A. Murphy, and S. Ritter. “closing the loop” in educational data science with an open source architecture for large-scale field trials. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 834–838, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852930>

millions of learners. Experiments at each scale have benefits and drawbacks, and there are important ways in which experiments at each scale can inform the design and implementation of experiments at the other scales. The authors’ presentation of the SEF concludes by laying out several key challenges for internet scale experiments. We briefly present their four posited challenges and review several (types of) educational data science/mining (EDS/EDM) studies that “close the loop” before describing the UpGrade open source architecture for conducting experiments in educational technologies at any of the SEF’s scales and how UpGrade helps educational technology developers address these challenges.

2. CHALLENGES FOR INTERNET SCALE EXPERIMENTS

Stamper et al. [18] lay out four key challenges for internet scale educational experiments. The first challenge is attracting a large user-base to the learning platform on which one would like to conduct such experiments. While an important issue, we leave the much broader discussion of large-scale adoption of educational technologies and software for learning for another day and assume that a researcher seeking to “close the loop” already is satisfied that they have a sufficiently large and diverse user-base to answer their research questions.

The second challenge they suggest, in the context of an educational game deployed via the Brainpop.com platform, is “instrumenting software for generating data logs that measure player performance, learning, and engagement” [18]. This is another broad, general challenge for developing software for learning, whether an educational game, intelligent tutoring system (ITS), or other type of learning software. Well-instrumented learning software provides insights about learners and their learning process, behaviors, engagement, and related facets of their learning experience that are the purview of nearly all work published at Educational Data Mining and related venues. We assume that readers and platform developers already recognize the importance of generating meaningful data from their learning platforms if they seek to run studies that close the loop or similar internet scale educational experiments.

Features of UpGrade target the third and fourth challenges raised by Stamper et al. [18]. The third challenge they pose is “the configuration of the software to allow for experimental designs” [18]. Being able to run experiments within a piece of software requires that different variants or instances of elements within an application’s design space can be instantiated and deployed in software, abstracted in such a way that the software can deliver

users to particular experiences based on their condition assignment. The software must also have some way of assigning users to particular experimental conditions. While variants within a target application's design space must still be created, UpGrade serves as an enabling technology to make experimental management (e.g., handling complexities of random assignment and tracking of users' condition assignments) less onerous for technology developers. One way of viewing UpGrade is as an enabling technology that makes A/B testing or experimentation a relatively simple matter of instrumenting target applications by implementing appropriate application programming interface (API) "hooks" to UpGrade rather than having to implement complex experimental management logic within the target application itself. Just as learning software developers increasingly understand the importance of instrumenting their software for high quality learning data collection, UpGrade may serve to further the goal that basic instrumentation in learning software will allow for rigorous learning engineering, involving experimentation and testing of new content, software features, instructional approaches, and other factors of interest to both researchers and technology developers.

The fourth challenge is two-fold, as Stamper et al. [18] note that "researchers increasingly face the challenge of making use of tens of thousands of subjects in an efficient manner" while also pointing out that some experiments may create inconsistent user experiences. While the authors view inconsistent user experiences as a potential catalyst for reducing overall participation, we take inconsistent user experiences as a more fundamental potential barrier to delivering internet scale experiments. Educational technology providers might reasonably refuse to deploy experiments at all if there are potential scenarios in which unsatisfying, inconsistent, or disengaging user experiences are likely to result.

Enabling technologies for large-scale educational experiments like UpGrade ought to deliver flexible options and capabilities to researchers and developers to deploy complex, substantive experiments and experimental designs in ways that still maintain high quality, consistent learning experiences. By illustrating practical examples of how studies that "close the loop" based on EDS/EDM insights might noticeably affect (or not) the learning experiences of real K-12 students and teachers, we motivate how UpGrade helps to meet many of the challenges raised by internet scale experiments.

3. "CLOSING THE LOOP"

We consider two kinds of potential "close the loop" studies drawn from literature in EDS/EDM and ITS research to provide a simple illustration of the types of considerations educational technology providers might make in delivering internet scale field trials or experiments to users in settings like K-12 classrooms. One case is intended to illustrate a situation in which a relatively simple, user-level random assignment study is unlikely to raise any concerns about consistency of learner experience while the other illustrates some potential concerns about consistency that might arise. In the section that follows, we describe important features and affordances of the UpGrade architecture in more detail to show how it enables researchers and educational technology developers to deploy experiments that address these concerns about consistency as well as provide other options for delivering high quality experiments for learning engineering.

The overarching goal of "closing the loop" with experiments is to evaluate whether observed improvements in (usually statistical) outcomes like prediction accuracy of particular models translate

into improved learning outcomes for students in a target system. Learning outcomes of interest in target systems might include efficiency of practice, time to skill mastery, or gains in performance from a pre-test measure to a post-test measure, among others.

Our example "close the loop" studies are related to how data-driven modeling of learner performance informs the specification and parameterization of so-called knowledge component (KC; or skill) [10] cognitive models frequently used within ITSs like Cognitive Tutor/MATHia [14] or in tutors built with tools like the Cognitive Tutor Authoring Tools [2].

A bevy of research in EDS/EDM and related areas (e.g., [3, 15, 19]) consider different approaches to fitting the parameters of KC models deployed in ITSs, typically within the four-parameter framework of Bayesian Knowledge Tracing [6], holding the set of KCs that appear to a learner during the learning experience fixed.

Contrast (1) studies that close the loop by contrasting *two or more sets of parameter estimates for the same KC model* with (2) studies that contrast two or more *different specifications of a KC model*, of which there are several examples in the literature (e.g., [11]). The second type of experiments are typically supported by semi-automated, data-driven techniques (e.g., Learning Factors Analysis [5]), various types of task analysis (e.g., [4]), or more recent multi-method approaches for "design loop adaptivity" [1, 8]. These techniques are used to re-evaluate the underlying KC model that drives the ITS's adaptive learning and determine how the specification of the KCs themselves might better represent what a student is learning (not just the performance parameters related to their learning) as they practice in the ITS.

In the first "parameter estimation" experiment in an ITS like MATHia, one or more experimental conditions and a control condition each have the same "skillometer" or dashboard display for students to see their progress toward KC mastery. Similarly, the same KCs or skills reported to teachers in their classroom analytics. The only difference in a parameter estimation experiment is likely to be exceedingly subtle, in that there are different parameters for subsets of KCs in each condition. In the latter "specification" experiment, the control condition and one or more experimental conditions vary in what KCs constitute the set of KCs used to drive adaptation for the topic as well as *what is displayed to students and their teachers*.

In the hypothetical parameter estimation study, individual random assignment is likely to be a reasonable choice for the researcher running the experiment. If the ITS, for example, implements some form of mastery learning (e.g., [17]), then students will be accustomed to receiving different amounts of practice on KCs they encounter within different topics in the ITS. Different parameterizations for different students in the same classroom are *not likely to lead to drastically different perceptions of the learning experience* for students. Nevertheless, if the experiment is successful, one or more parameterizations may lead to more efficient practice or provide students with additional practice that they need to achieve mastery. Teachers' experience of using analytics and reports are likely to be nearly indistinguishable across the different parameterizations.

Though perhaps still subtle, KC model specification studies are more likely to create inconsistencies in K-12 learners' and instructors' experiences in an ITS if they were to be deployed within, for example, the same classroom, or even to all of the classrooms of the same teacher. Changes in the KC model specification may also be accompanied by design differences in the

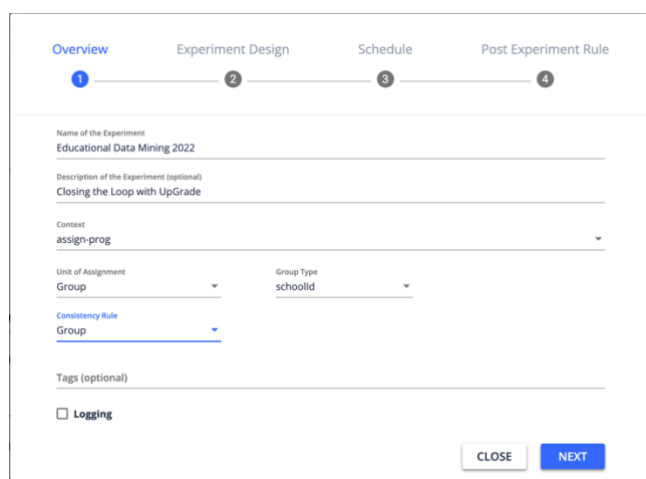
tasks presented to students (e.g., as in [8, 18]), presenting further opportunities for noticeable differences in learner experiences to emerge.

Generally, the contrast we seek to illustrate is between any type of relatively “stealthy” experiment with more “conspicuous” and/or visually salient experiments. More stealthy examples, like the hypothetical experiment that only manipulates parameter estimates, are those in which differences in the learner experiences are subtle and likely to be unnoticed between conditions. Experiments are likely to be more conspicuous when they seek to test substantive differences in learning experiences and may have more easily-discernable variations, as in the “specification” experiment.

More conspicuous differences might reasonably be evaluated between instructional methods, encouraging students to adopt different problem-solving strategies for the same topic, and contrast along any number of a wide variety of instructional decisions content designers must make [9]. Many such differences between experimental conditions are easily perceived by students and teachers. If condition assignment is not thoughtfully considered (e.g., by considering a group-level assignment), students may realize that they are receiving a different experience compared to other learners in their classrooms, leading to unanticipated changes in learning patterns. Teachers might easily become overwhelmed by the need to support different learning experiences for the same topic, keep track of differences in analytics and reporting, and other potential inconsistencies. In what follows, we describe how the UpGrade platform can help educational technology developers and researchers not only deploy internet scale educational experiments but also do so in ways that handle many of the challenges that arise in such deployments in real world settings in which learning takes place.

4. UPGRADE

UpGrade¹ [16] is an open source software architecture (available via GitHub²) intended to lower barriers to learning engineering and enable internet scale experiments (also sometimes referred to as “A/B tests” or randomized field trials) that test substantive changes to learning experiences in settings like K-12 classrooms while preserving consistent, high-quality learning experiences for end-users.



The screenshot shows the 'Overview' step of a four-step process for creating a new experiment. The steps are: Overview (1), Experiment Design (2), Schedule (3), and Post Experiment Rule (4). The 'Overview' step is currently active. The form includes the following fields and options:

- Name of the Experiment: Educational Data Mining 2022
- Description of the Experiment (optional): Closing the Loop with UpGrade
- Context: assign-prog (dropdown menu)
- Unit of Assignment: Group (dropdown menu)
- Group Type: schoolid (dropdown menu)
- Consistency Rule: Group (dropdown menu)
- Tags (optional):
- Logging
- Buttons: CLOSE and NEXT

Figure 1. Screenshot of the “overview” of an experiment in the UpGrade user interface for creating a new experiment

We consider several of the major affordances provided by UpGrade, several of which are illustrated in the screenshot of Figure 1 that shows the preliminary “overview” of an UpGrade experiment as it is being specified. Group random assignment, consistency rules, post-experiment rules, and user segmentation are features of experiments that help to ensure that consistent learning experiences are delivered during experiments. Researchers determine how best to set these options as a part of designing experiments in UpGrade.

4.1 Group Random Assignment

While UpGrade has logic for assigning experimental conditions on an individual-student or user basis, the ability to randomly assign conditions by group (e.g., at the level of classrooms within a K-12 school) enables researchers to conduct internet scale experiments in educational software products that are both deployed at scale and used in authentic classroom contexts. In educational settings, it may be undesirable for students within the same group (e.g., by class, teacher, school district, or some other grouping) to be assigned different conditions within the same experiment, particularly if such conditions involve conspicuous or salient visual changes (e.g., different “skillometers” displays and teacher analytics in the KC model specification experiment illustrated in §3) or substantively different models of instruction. UpGrade manages coherence of learning experiences by group as well as anomalies that may arise in group membership, enabling researchers to specify how an experiment should behave if a student switches classes or is in multiple classes simultaneously.

4.2 Consistency Management

The second way in which UpGrade helps deliver consistent experiences is via associating deployed experiments with consistency rules that govern how users are treated for inclusion/exclusion in an experiment who have already encountered pieces of instructional content or other design features that are included in experiments. This is particularly useful when instructional content is delivered via adaptive software in which self-paced progress is often a crucial feature of the learning experience; in such software students may reach the content of interest at different times. If a student in a class encounters the content of interest earlier (or later) than their fellow students, consistency management can specify whether condition assignment binds more strongly to group membership or individual students.

4.3 Post-Experiment Rules

A “post-experiment rule” is a parameter that manages delivery of experimental conditions once an experiment has stopped running, but students may still interact with the target content in the educational application. Researchers may wish to have a “winning” condition be delivered to subsequent students who encounter the content of interest, or may wish to maintain the condition assignment weighting even if the experiment has ended. For example, if a study using UpGrade runs from September-November of a school year, but a student goes back to review content in preparation for an end-of-semester exam in December, a post-experiment rule can specify whether that student should receive the same experimental condition they were originally assigned, or whether they are permitted to experience a default or other condition.

¹ <https://www.upgradeplatform.org/>

² <https://github.com/CarnegieLearningWeb/UpGrade>

4.4 User Segmentation

Another challenge when conducting internet-scale experiments in real-world classrooms is that researchers may not necessarily want or need to target all members of a population. Segmentation and the ability to pre-define include/exclude lists empowers researchers to target experimental interventions to groups of interest, at the level of interest: e.g., middle schools, 7th grade, specific districts, or even geographic regions. Similarly, districts or schools can fully “opt-out” and join a global exclude list without impacting research at scale conducted by an educational technology company.

While learning platforms often do not automatically collect data about individual learner demographics, when such data are available (or when such data are available at an aggregated level like that of particular schools), user segmentation is likely to also play an important role in better understanding what works for particular sub-populations of students, helping researchers close the loop in ways that promote inclusion and equity across diverse populations of learners.

4.5 Monitoring Metrics

UpGrade enables researchers running experiments to monitor their progress with respect to “enrolled” users (i.e., those users who have encountered relevant content and been assigned to a condition) as well as those who have been excluded. In addition, APIs are available for target learning applications to send specific metrics (e.g., time to complete a particular piece of content) of interest to UpGrade’s monitoring dashboards for real-time progress monitoring of experimental progress. The screenshot in Figure 2 illustrates a particular case of an experiment with two conditions in the UpGrade platform, displaying enrollment data over time and by experimental condition.



Figure 2. Screenshot of monitoring dashboard in UpGrade, showing enrollment metrics over time and by condition, for two experimental conditions, entitled “ddi_control” and “pre_ddi_variant,” where “ddi” stands for “data driven improvement” of particular content in Carnegie Learning’s MATHia platform.

4.6 Support for Diverse Experimental Designs

UpGrade currently supports relatively simple experimental designs, including weighted random assignment to two or more conditions (see “weight” column near the bottom of the screenshot in Figure 2) as well as within-subject, factorial, and partial-factorial designs. UpGrade developers have a roadmap for implementing additional designs including multi-arm bandits and stepped-wedge designs in the near future. More sophisticated designs and those

incorporating adaptive experimentation could be valuable contributions from the EDM and allied communities. This leads to our call for contributors and integrators.

5. CALL FOR CONTRIBUTORS & INTEGRATORS

Our goal in the present paper has been to introduce UpGrade to the EDM community. We hearken back to the “rise of the super experiment” and seek to build awareness of challenges often raised by internet scale, experimental “close the loop” studies and how UpGrade presents a solution that can be integrated within an existing or emerging educational technology product to help overcome those challenges. Efforts at the intersection of EDS/EDM and the emerging field of learning engineering rigorously seek to establish causal links between data-driven insights that inform improvements in educational technologies and practical learning outcomes resulting from the use of these technologies. We applaud efforts like E-TRIALS [12], MOOClets [13], and Terracotta³ that aim to provide similar support for rigorous experimental or A/B testing of learning innovations within particular contexts (e.g., E-TRIALS within ASSISTments [7], TerraCotta within the Canvas Learning Management System). UpGrade can be integrated into existing or new learning applications and technologies to similarly drive rigorous data-driven improvements to learning platforms.

Educational technology developers must still (as ever) address challenges to attracting large and diverse user-bases, instrumenting their technology to capture rich learning data, and appropriately abstracting features and content in their systems so that different learning experiences can be delivered to learners (Challenges #1-2 and part of the third challenge described by [18]). However, UpGrade removes many of the barriers imposed by the challenges of large-scale experimental management in real-world learning settings. The educational technology developer need only implement software instrumentation that calls UpGrade’s API to determine which alternative learning experience (if any) ought to be delivered to a particular user, given characteristics of that user that the target system “knows” about (e.g., via communication with a rostering system) such as the class or school in which the user is learning. The target system for experimentation can also implement UpGrade’s API to provide metrics for monitoring experiments as they proceed. UpGrade handles complex logic of managing condition assignments and dealing with real-world complexities that inevitably arise in settings like K-12 schools.

As an open source platform, developers can contribute new functionality and features to the codebase for the future benefit of all integrators and researchers using platforms that integrate with UpGrade. For example, code might be contributed to build connections to deliver A/B tests in different LMSs and implement appropriate APIs to have metrics for monitoring delivered to UpGrade. Support for new types of experimental designs and algorithms for adaptive experimentation are also a natural place for future development. We welcome such new contributions from the EDM community as well as from the broader educational technology and learning engineering communities.

UpGrade has already been used to deliver experiments to tens of thousands of learners in a math game similar to that targeted in the paper that introduces the SEF [18]. A number of experiments are currently deployed in Carnegie Learning’s MATHia, and new experiments will be deployed in the coming months using

³ <https://terracotta.education/>

UpGrade. These experimental field trials will close the loop between data-driven improvements to facets of learning experiences like KC models as well as personalization and motivational features, and we look forward to presenting those results in the near future. We encourage educational technology developers to consider integrating UpGrade into their platforms to enable rigorous, iterative learning engineering improvements and platform-enabled educational research.

6. ACKNOWLEDGMENTS

The research reported here was supported by the Institute of Education Sciences, U.S. Department of Education, through Grant R305N210045 to Carnegie Learning, Inc. The opinions expressed are those of the authors and do not represent views of the Institute or the U.S. Department of Education. The first and third authors are also supported by the National Science Foundation under award The Learner Data Institute (Award #1934745). The opinions, findings, and results are solely the authors' and do not reflect those of the National Science Foundation.

7. REFERENCES

- [1] Aleven, V., McLaughlin, E.A., Glenn, R.A., Koedinger, K.R. 2017. Instruction based on adaptive learning technologies. In *Handbook of Research on Learning and Instruction*, 2nd Edition. Routledge, New York, 522-560.
- [2] Aleven, V., Sewall, J., McLaren, B. M., & Koedinger, K. R. 2006. Rapid authoring of intelligent tutors for real-world and experimental use. In *Proceedings of the 6th IEEE International Conference on Advanced Learning Technologies*. ICALT 2006. IEEE Computer Society, Los Alamitos, 847-851.
- [3] Baker, R.S., Corbett, A.T., Gowda, S.M., Wagner, A.Z., MacLaren, B.M., Kauffman, L.R., Mitchell, A.P., Giguere, S. 2010. Contextual Slip and Prediction of Student Performance After Use of an Intelligent Tutor. In *Proceedings of the 18th Annual Conference on User Modeling, Adaptation, and Personalization*, 52-63.
- [4] Baker, R. S., Corbett, A.T., and Koedinger, K.R. 2007. The difficulty factors approach to the design of lessons in intelligent tutor curricula. *Int. J. Artif. Intell. Educ.* 17(4), 341-369.
- [5] Cen, H., Koedinger, K.R., and Junker, B. 2006. Learning factors analysis: A general method for cognitive model evaluation and improvement. In *Intelligent Tutoring Systems 2006*. ITS 2006. Springer-Verlag, Berlin, 164-175.
- [6] Corbett, A.T., Anderson, J.R. 1995. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Model. User-Adap.* 4, 253-278.
- [7] Heffernan, N.T., and Heffernan, C.L. 2014. The ASSISTments Ecosystem: Building a platform that brings scientists and teachers together for minimally invasive research on human learning and teaching. *Int. J. Artif. Intell. Educ.* 24, 470-497.
- [8] Huang Y., Aleven V., McLaughlin E., and Koedinger K. 2020. A general multi-method approach to design-loop adaptivity in intelligent tutoring systems. In *Artificial Intelligence in Education 2020*. AIED 2020. LNCS, vol 12164. Springer, Cham, 124-129.
- [9] Koedinger, K.R., Booth, J.L., and Klahr, D. 2013. Instructional complexity and the science to constrain it. *Science* 342(6161), 935-937.
- [10] Koedinger, K.R., Corbett, A.T., and Perfetti, C. 2012. The knowledge-learning-instruction framework: Bridging the science-practice chasm to enhance robust student learning. *Cognitive Science* 36(5), 757-798.
- [11] Liu, R., and Koedinger, K.R. 2017. Closing the loop: Automated data-driven cognitive model discoveries lead to improved instruction and learning gains. *Journal of Educational Data Mining* 9(1), 25-41.
- [12] Ostrow, K., and Emberling, R. 2020. E-TRIALS: A web-based application for educational experimentation at scale. In *Proceedings of the First Workshop on Educational A/B Testing at Scale*. EdTech Books.
- [13] Reza, M., Kim, J., Bhattacharjee, A., Rafferty, A.N., & Williams, J.J. 2021. The MOOClet Framework: Unifying experimentation, dynamic improvement, & personalization in online courses. In *Proceedings of the 8th ACM Conference on Learning at Scale*. L@S 2021. ACM, New York, NY, 15-26.
- [14] Ritter, S., Anderson, J.R., Koedinger, K.R., and Corbett, A.T. 2007. Cognitive Tutor: applied research in mathematics education. *Psychon. B. Rev.* 14, 249-255.
- [15] Ritter, S., Harris, T.K., Nixon, T., Dickison, D., Murray, R.C., Towle, B. 2009. Reducing the knowledge tracing space. In *Proceedings of the 2nd International Conference on Educational Data Mining* (Cordoba, Spain, 2009), 151-160.
- [16] Ritter, S., Murphy, A., Fancsali, S. E., Fitkariwala, V., Patel, N., and Lomas, J. D. 2020. UpGrade: An open source tool to support A/B testing in educational software. In *Proceedings of the First Workshop on Educational A/B Testing at Scale*. EdTech Books.
- [17] Ritter, S., Yudelson, M., Fancsali, S.E., and Berman, S.R. 2016. How mastery learning works at scale. In *Proceedings of the 3rd ACM Conference on Learning at Scale* (April 25 - 26, 2016, Edinburgh, UK). L@S 2016. ACM, New York, NY, 71-79.
- [18] Stamper, J.C., Lomas, D., Ching, D., Ritter, S., Koedinger, K.R., Steinhart, J. 2012. The rise of the super experiment. In *Proceedings of the 5th International Conference on Educational Data Mining* (Chania, Greece, June 19-21, 2012). EDM 2013. International Educational Data Mining Society, 196-199.
- [19] Yudelson, M., Koedinger, K., Gordon, G. 2013. Individualized Bayesian Knowledge Tracing models. In *Proceedings of 16th International Conference on Artificial Intelligence in Education* (Memphis, TN). AIED 2013. LNCS vol. 7926. Springer-Verlag, Berlin / Heidelberg, 171-180.

Estimating the causal effects of Khan Academy MAP Accelerator across demographic subgroups

Phillip Grimaldi, Kodi Weatherholtz, and Kelli Millwood Hill
Khan Academy
Mountain View, CA, USA
efficacy-team@khanacademy.org

ABSTRACT

As educational technology platforms become more and more commonplace in education, it is critical that these systems work well across a diverse range of student sub-groups. In this study, we estimated the effectiveness of MAP Accelerator; a large-scale, personalized, web-based, mathematics mastery learning platform. Our analysis placed a particular focus on students from historically under-resourced groups. Our sample comprised 181K students in grades 3-8 from 99 school districts across the United States, 52% of whom attended schools where the majority of the students are eligible for free or reduced price lunch (a proxy for high-poverty schools). Using a combination of system logs/platform telemetry data, standardized assessments, and publicly available school/district data, we estimated the causal effect of a year-long supplemental math intervention on students' standardized mathematics growth outcomes via a quasi-experimental design with controls and a modification of the difference-in-differences framework. Students who used the platform as recommended (30+ min/wk) during the (COVID-19 disrupted) 2020-2021 school year had math growth scores 0.26 standard deviations higher on average than similar students who used the platform for <15 min/week. Further, positive benefits of the platform were observed across all genders, race/ethnic groups, and school poverty levels, but were not as large for districts with greater than 20% English language learners. Further analysis revealed that these students were predominantly Hispanic, and tended to make less progress on fewer skills than their peers given the same amount of practice time. We discuss the implications of these findings, and potential avenues to ensure more equitable outcomes for these students in the future.

Keywords

equity, causal inference, mastery learning, Khan Academy, English Language Learners

1. INTRODUCTION

P. Grimaldi, K. Weatherholtz, and K. M. Hill. Estimating the causal effects of Khan Academy Map Accelerator across demographic subgroups. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 839–846, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852932>

Perhaps the most natural question to ask concerning any educational intervention is “does it work?” Indeed, the *What Works Clearinghouse* has reviewed over 11,000 studies evaluating the efficacy of various programs and technologies [26]. Overwhelmingly, these studies are concerned with establishing how well the intervention works *in general*. Of course, the results are inherently limited to the student population in the study, and generalizing the results is not always straightforward. In the U.S., student populations are becoming increasingly diverse [13], with students coming to the classroom with differing skills, backgrounds, and needs. Moreover, there is a growing awareness of educational inequities across the student population coupled with concerns about systemic bias. Even if a study sample is representative of national norms, it is unreasonable to expect that the expected effect would work the same for every subgroup within the sample. Thus, it is critical that we move from asking only “does it work?” to a more specific “for whom does it work?”

In this study, we estimated the effectiveness of Khan Academy's MAP Accelerator, a large-scale, personalized, web-based, mathematics mastery learning platform. Like many other such studies, we used performance on a standardized test, NWEA MAP Growth™, to measure the impact of the intervention. We utilized statistical controls to rule out the influence of confounding variables when possible, and applied difference-in-differences to control for unobserved confounding as well. Critically, we also repeated the analyses across a variety of student sub groups (grade, gender, ethnicity, socioeconomic status, English language proficiency) in order to determine whether the software worked equally well for these groups.

2. BACKGROUND

2.1 NWEA MAP Growth

MAP Growth is a computerized, adaptive, standardized assessment designed to measure and track student mathematics, reading, language usage, and science [18]. For the purposes of this report, we are primarily interested in the mathematics assessment. MAP Growth is designed to be used as an interim assessment, allowing educators to monitor progress and tailor instruction in advance of a summative assessment. The assessment is typically administered three times per year—Fall, Winter, and Spring—with an optional summer administration. During the Fall 2020 administration, the MAP Growth test was completed by nearly 4.4 million students in the United States [14].

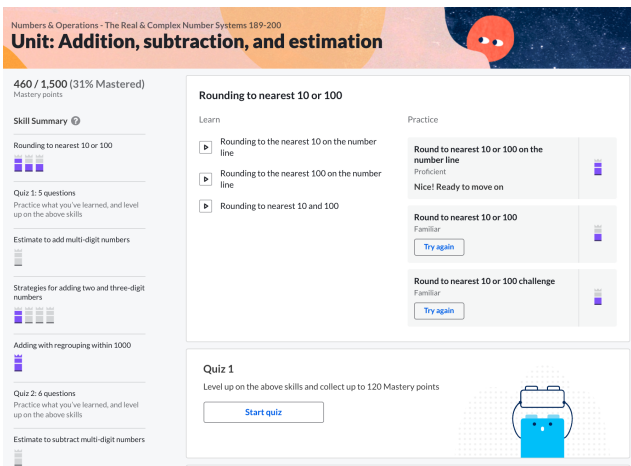


Figure 1: Screenshot of MAP Accelerator unit page. Students can learn, practice, quiz, and view mastery states from this page.

MAP Growth is based on the Rasch Model [2], which measures latent student ability on the tested domain. The assessment provides a “RIT” score (short for “Rasch Unit”), which is a linear transform of the θ estimates from the Rasch model. RIT scores typically range from 100 to 350. Given the adaptive nature of the assessment, the RIT scale is not grade dependent. Instead, a student’s RIT score improves incrementally across school years, affording the ability to track progress longitudinally. RIT scores can be categorized into different “bands”, which describe the expected range for students at that grade level. Scores are also offered for four high level subscales: 1) Geometry, 2) Number & Operations/The Real & Complex Number Systems, 3) Operations & Algebraic Thinking, and 4) Measurement & Data/Statistics & Probability.

2.2 MAP Accelerator

MAP Accelerator is a web-based supplemental math learning tool designed to be used in schools that are also using the MAP Growth assessment. At the beginning of the school year, a student’s subscores from their fall MAP Growth assessment are sent to MAP Accelerator, which then constructs for them a set of supplemental instruction and practice content. For example, if a student’s RIT score for Geometry was in the 176-188/Grade 3 range, their lessons in Geometry would be aligned to that level.

A full description of MAP Accelerator’s functionality is beyond the scope of this paper. However, the key take away is that students can learn, practice, or quiz on content of their choosing (see Figure 1). Learn refers to study activities such as explanatory videos, worked examples, and articles. Practice refers to solving interactive math practice problems on specific skills[8]. Practice opportunities typically provide hints to students, as well as immediate elaborative feedback[21] after failed attempts. Quizzing is similar to practice, except provides an opportunity for students to self-test in a mixed skill environment.

MAP Accelerator also implements concepts from *mastery*

learning [15]. The idea behind mastery learning is that students should work at their own pace to master content before advancing on. MAP Accelerator monitors student performance on practice blocks, quizzes, and mastery challenges and provides feedback on their estimated level of mastery (attempted, familiar, proficient, or mastered). While MAP Accelerator allows students to choose their own path, the mastery feedback allows for easy selection of topics that need the most help.

Importantly, MAP Accelerator is intended to be used as a supplement and not a primary source of instruction. Teachers are encouraged to have their students use MAP Accelerator for at least 30 minutes of focused learning time per week—a dosage level that is meant to be reachable with one dedicated class period per week.

2.3 Impact of COVID-19 Pandemic

MAP Accelerator was first broadly released during the 2020-2021 school year. This was the peak of the COVID-19 pandemic and prior to wide spread availability of vaccines. Nearly two-thirds of U.S. students were enrolled in distance learning formats during the Fall 2020 school year [13]. In addition to academic life, home life was disrupted as well. Many parents were losing their jobs or shifting to at home work. Thus, many students had to complete the school year jockeying for internet access and a quiet place to study. Of course, the degree of disruption would obviously depend on socioeconomic-status of the household and available resources.

Given the extreme disruption to academic and home life, it is not surprising then that student achievement measures would drop. Lewis et al. [17] reported that overall Math RIT scores in Fall of 2020 were considerably lower than pre-pandemic levels. Moreover, the amount of growth from Fall 2020 to Spring 2021 was lower than historical averages. Thus, not only did students start the school year behind where they should have been, they grew at a slower pace than they would have been expected to pre-pandemic.

Lewis et al.[17] also found that the negative effects of the pandemic were not observed equally across all ages or demographic groups. First, younger students (grades 3-5) were more negatively affected than older students (grades 6-8). Second, pandemic related declines were more pronounced in students that attended schools with a high proportion of students on free or reduced lunch. Finally, students from historically under-served groups (e.g., Hispanic/Latino, Black-/African American, American Indian/Alaskan Native) were more negatively affected than White and Asian students. In sum, the pandemic appears to have hit vulnerable populations the hardest, and exacerbated pre-existing inequities and achievement gaps.

3. METHODS AND RESULTS

3.1 Data and Sample Summary

In the sections below, we describe some of the data that was available to us, as well as any feature engineering we did. Data from Khan Academy and NWEA MAP Growth tests were privately shared with the researchers for this study. We also made use of publicly available government data from the

National Center for Education Statistics (NCES). All data and analyses are limited to U.S. Schools that adopted MAP Accelerator for the 2020-2021 school year, and to students who took both a Fall and Spring administration of the test. A summary table of the data is available in Appendix A.

Several measures were taken to protect the privacy of students and teachers in this study. All data were deidentified, and personally identifiable information was removed and replaced with electronically encrypted hashed fields. Key files to decrypt the hashed fields were not shared with the researchers. Deidentified data was stored on encrypted and password protected servers. Only researchers directly involved with the project were permitted access to the data. Finally, all data was used in accordance with MAP Accelerator Supplemental Terms and Conditions, which was agreed to by participating districts.

3.1.1 School Data

For each student, we were provided with `district_name`, `school_name` (school the student started the year enrolled in), and `math_teacher` (unique math teacher identifier).

Using this information, we were able to merge publicly available school level variables from the NCES. For each school, we obtained the percent of students reported to be on Free or Reduced Price Lunch (FRL), a commonly used indicator of socioeconomic status[25]. From this, we created a `school_frl_level` variable by binning schools into one of four groups: High FRL ($\geq 75\%$), Med FRL ($> 25\%$, $< 75\%$), Low FRL ($\leq 25\%$), or Unknown. We also obtained the percent of English Language Learners at the district level (ELL). From this we created a `district_ell_level` variable by binning into four groups: High ELL ($\geq 20\%$), Med ELL ($\geq 5\%$, $< 20\%$), Low ELL ($< 5\%$), and Unknown.

The key point to note about the school level data is that 52% of the sample came from High FRL schools. The overall sample skews more towards high poverty schools than would be expected from a random sample of US schools (24%)[24].

3.1.2 Demographic Data

For each student, we were provided with their `gender` (Male, Female), `grade` (3 - 8), and `ethnic_group` (American Indian or Alaska Native, Asian, Black or African American, Hispanic or Latino, Native Hawaiian or Other Pacific Islander, White, Multi-ethnic, Not specified or Other).

A key point to note about the demographics of this sample (see Appendix A) is that the ethnicity breakdown is much different than national averages. In particular, Hispanic or Latino students are just as prevalent in the study sample as White students. Nationally we would expect around 47% White and 27% Hispanic or Latino [13].

3.1.3 MAP Accelerator Usage

For each student we obtained a summary of their total minutes using MAP Accelerator, filtered to the time period between when they took their Fall 2020 and Spring 2021 MAP Growth assessments. Further, only “on task” (i.e., not navigation) minutes working with math content specifically was used. Because the duration of the period between tests varied across students, we then divided the total minutes by

the number of weeks between the two assessments. Lastly, we created a categorical `usage_group` variable by binning the total minutes per week into four groups: *No use*, *< 15 min/wk*, *15-30 min/wk*, or *30+ min/wk*.

Only 5% of students reached the recommended level of 30+ min/wk, and only 9% reached the next highest level of 15-29 min/wk (see Appendix A for full table). We conducted an exploratory analysis to examine how much school level factors predicted an individual students level of usage. We fit a multi-level regression model predicting students’ total learning minutes on MAP Accelerator based on a single population-level intercept (global mean) and random intercepts for teacher, school and district. The random effect variance estimates are shown in Table 1. The intraclass correlation coefficient (ICC) indicated that 57% of the variance in students usage is explained by the environmental factors of teacher, school and district [12].

Table 1: Variance estimates from a multi-level regression model predicting users’ learning minutes on MAP Accelerator.

	Variance	Proportion of variance
math_teacher	38741.12	0.22
school_name	25034.16	0.14
district_name	35668.95	0.20
Residual	74729.04	0.43

^a ICC = 0.57

3.1.4 MAP Growth Data

NWEA provided several measures for each student, the critical ones being: `baseline_test_rit_score` (Fall 2020 RIT score), `endline_test_rit_score` (Spring 2021 RIT score), `subject` (Math or reading), and `conditional_growth_index` (population-normalized fall to spring gains).

The *Conditional Growth Index* (CGI) will be our primary outcome variable of interest, thus it warrants additional explanation. CGI is essentially a z-score of a students simple Fall to Spring growth relative to all other test takers. However, unlike a standard z-score, which is based on the population mean and standard deviation, the calculation is conditioned on the student’s starting RIT score, grade, and total instructional time between test events. CGI is then interpreted as the standard deviation change in RIT, relative to all other students who also started with the same RIT score (see [23] for more information on CGI calculation). An interesting aspect of CGI in the context of this study is that it utilizes pre-pandemic national norms. This means that CGI contextualizes a students’ observed growth relative to their expected growth prior to the pandemic. Concretely, a CGI of 0 does not mean that the student did not grow, but rather that they grew at a rate that would be expected for a similar student pre-pandemic.

We use CGI over simple growth for several reasons. The first is that it is a standardized metric that is easier to interpret without knowledge of the RIT scale. The fact that it is standardized against pre-pandemic norms also provides added context for interpreting the results for a COVID dis-

rupted school year. The second reason has to do with the unique properties of the RIT scale. In particular, typical growth rates are not consistent across grades[18]. Students in earlier grades tend to gain more from Fall to Spring than students in later grades. Even within a grade, students with higher RIT score may actually be expected to grow less than students with a lower starting RIT score – which is not a typical or intuitive pattern[5]. Moreover, the logic of the difference-in-difference framework hinges on the “parallel trends” assumption [11] which is that the treatment group should be expected to follow the same trend as the untreated group if no treatment had taken place. The fact that expected growth depends on Fall 2020 starting values means that examining only raw scores would likely violate the parallel trends assumption. Standardization with CGI overcomes this issue.

3.2 Design Overview

In general, this study will use a quasi-experimental, pretest-posttest nonequivalent groups design with controls. We leverage the natural variations in usage to infer the impact of using MAP Accelerator on gains in CGI, while controlling for covariates. Even with statistical controls, this design has the potential for confounding from unobserved variables. For example, students who received more parental support during the pandemic may have been more likely to succeed academically[10, 3] and more likely to use MAP Accelerator[19]. We account for unobserved confounding by using the logic of Difference-in-differences (DID)[6].

Difference-in-differences is a causal inference technique that compares an outcome measure before and after an intervention against a counterfactual. For example, the impact of a new traffic law could be evaluated by comparing traffic accidents before and after passing of the law both in the state that passed the law and in a neighboring state that did not. The logic is that the counterfactual (i.e., neighboring state) affords the ability to account for any changes in traffic accidents that could be attributable to other factors, such as bad weather. In our case we compare a low usage group (<15 min/wk) to a recommended usage group (30+ min/wk) on test scores before and after the intervention. We take the logic a step further by comparing the DID of two outcome measures, math and reading. Because MAP Accelerator is designed as a math intervention, we expect effects to be observed primarily in math outcomes. However, an unobserved confound such as parental support should also influence reading performance. DID provides a framework for removing the influence of these confounds. Finally, we add one more layer to DID by conducting a similar analysis across demographic subgroups in order to infer the effects within each of those groups.

3.3 Analysis 1

First we modeled `conditional_growth_index` for both math and reading using mixed effect regression models. We used the following fixed effects: `usage_group`, `gender`, `ethnic_group`, and `baseline_test_rit_score`. We also included the following random effects: `district_name`, `school_name`, `math_teacher`. Math and reading were modeled separately. The critical results are shown on Figure 2, which shows the corrected marginal means of CGI for each of the usage groups.

As we see on Figure 2, there was a positive relationship between usage levels of MAP Accelerator and math CGI. Students in the *No use* group showed the lowest CGI scores, and CGI improved with each usage level. Interestingly, we also see a similar pattern for reading. Given that we did not expect substantive effects of a math supplement on reading performance, this pattern is problematic as it suggests the existence of unobserved confounding in the usage groups. However, we do note that the estimated effects of MAP Accelerator usage were larger for math than for reading. Notably, students in the *30+ min/wk* group showed growth that was approximately 0.23 standard deviations higher than expected for math, but slightly below expected levels for reading. Thus, while there may be a degree of confounding, it does not fully explain the relationship between MAP Accelerator usage and performance. Nevertheless, it does make it difficult to determine how much of the true causal effect can be reasonably attributed to MAP Accelerator.

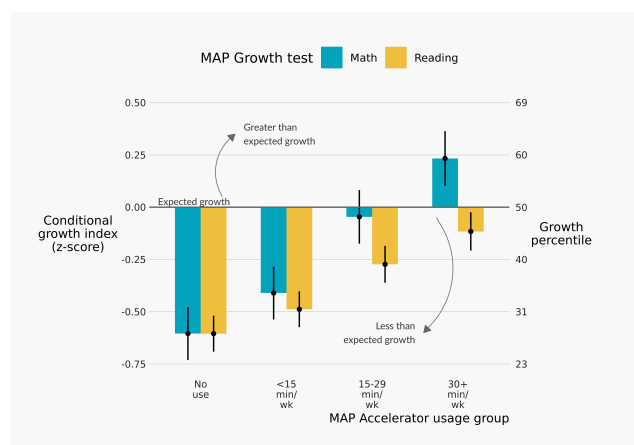


Figure 2: Estimated Conditional Growth Index as a function of MAP Accelerator usage group and test domain. Error bars are 95% CI.

3.4 Analysis 2

In order to account for unobserved confounds, and to estimate the effects across demographic subgroups, we applied the DID approach as previously discussed. We fit a mixed effect regression model using the `lme4` package in R[1] and the specifications on Listing 1. The interaction terms model the various DID effects. The overall main effect was modeled as a two-way `usage_group * subject` interaction. Similarly, the effects for individual subgroups were modeled as three-way interactions (e.g., `usage_group * subject * grade`). The three-way interactions allow us to estimate the causal effects of MAP Accelerator across subgroups.

After fitting the model, we conducted post-hoc contrasts to extract the DID effects of interest, using the `emmeans` package[16]. We focus on contrasting the two highest usage groups (*30+ min/wk*, *15-29 min/wk*) against the lowest usage group (*<15 min/wk*), and excluded the *No use* group from further analysis. On the surface, the *No use* group seems a natural reference point, however we noted fairly large differences in pre-intervention measures between the *No use* group and other groups, in some cases exceeding


```

lmer(conditional_growth_index ~
  usage_group * subject *
  (grade + gender + ethnic_group +
  school_frl_level +
  district_ell_level +
  math_baseline_test_percentile) +
  (1 | math_teacher) +
  (1 | school_name) +
  (1 | district_name),
  data = ...,
  control = lmerControl(
    optimizer="bobyqa",
    optCtrl=list(maxfun=2e5)),
  REML = FALSE)

```

Listing 1: Full model specification for difference-in-differences effects.

the maximum standardized mean difference recommended by the What Works Clearinghouse [27]. Moreover, the total lack of usage may signal a substantive difference in and of itself. For example, students in the *No use* group may have faced technological barriers, such as limited computer or internet access, that prevented them from using the tool in the first place. Note that overall usage in the <15 min/wk group was very low (Median 1.7 Hours over the entire year). Thus, we would not expect usage at these levels to have meaningful benefits on growth over an entire school year.

The critical results are shown on Figure 3. We will focus first on the overall effect, as it provides a nice explanation of how the DID are calculated. The panel A of Figure 3 shows the marginal means of CGI for each of the usage groups on both math and reading. Presented this way, we see a clear interaction – CGI improves incrementally with each usage group, but the improvement is larger for math than reading. The main DID is shown on panel B of Figure 3. These estimates are the less-contaminated causal effects of using MAP Accelerator on math CGI. The effect was larger for the *30+ min/wk* group than the *15-29 min/wk* group (+0.26 vs. +0.15). The direction of this difference is consistent with a causal effect of using MAP Accelerator.

The panel C of Figure 3 shows the DID effects broken down by various subgroups. In general, the pattern of *30+ min/wk* students showing larger effects than *15-29 min/wk* students held across most of the groups. However, while the magnitude of these gaps did fluctuate, so too do the confidence intervals, ultimately preventing any firm conclusions to be drawn regarding the relative effectiveness of 30+ min/wk vs. 15-29 min/wk within each group. Regardless, both usage levels were consistently positive and greater than 0 across subgroups. In particular, there were no meaningful differences across grades, gender, ethnicity¹, or school FRL.

Figure 3 did reveal that students from high ELL districts did not experience the same benefits from their use of MAP Accelerator as their peers. We conducted some post-hoc ex-

¹We did not report results from Am. Indian/ Alaska native due to having too small a sample sizes. Not specified/other and multi-ethnic were also excluded because these labels were not specific enough for the results to provide practical utility.

ploratory analyses to learn more about these students. High ELL districts were demographically distinct from the overall sample, with a higher proportion of Hispanic or Latino students (64%), consistent with national norms[13]. The vast majority of MAP Accelerator usage in High ELL districts was in English (99.7%). Using additional MAP Accelerator data, we computed a rough estimate of learning efficiency by estimating the proportion of skills leveled up over the school year. Students from high ELL districts tended to level up an average of 5.3 (SE = .06) skills per hour, compared to 7.2 (SE = .03) for mid ELL and 8.9 (SE = .07) for low ELL.

4. DISCUSSION

We applied difference-in-differences with controls to estimate the causal effect of MAP Accelerator on NWEA MAP Growth assessment across a wide range of student subgroups. In general, we observed positive effects of MAP Accelerator, with an overall effect size of +0.26 for students who used at the recommended levels of usage. Effect sizes of that magnitude are fairly common in educational technology research [22, 9]. These results provide additional support for mastery learning and the importance of tailoring instruction to students ability levels[15]. In future analysis, we plan to conduct more targeted analyses to determine which specific features of MAP Accelerator resulted in the most benefits.

While the finding that use of MAP Accelerator improved MAP Growth scores is good news, such effects would be less welcomed if it only helped some students. Fortunately, the benefits of MAP Accelerator appeared mostly consistent across grade, ethnicity, and school FRL. These results are important, especially given the context of COVID-19 and known existing equity gaps. As noted earlier, Lewis et al. [17] found that the pandemic more negatively affected the change in MAP Growth scores of younger students, students from high FRL schools, and Black and Hispanic or Latino students. Here we see indication that the pattern was halted, at least for students who used MAP Accelerator at recommended levels. An exception was that students from high ELL schools did not appear to benefit as much from their use of MAP Accelerator as other students.

Why didn't students from High ELL districts benefit from their use of MAP Accelerator? Prior research on ELL students has shown that learning new content in a language other than their primary language can be a source of cognitive overload [20]. The fact that these students also progressed on MAP Accelerator at a slower pace is consistent with this explanation. Other research has found that programs that allow students to receive instruction in their primary language is more effective relative to comparison groups[4]. Interestingly, students had the option to switch MAP Accelerator to their preferred language, but this was extremely rare. Students may have not been aware of this feature, or were instructed not to use it. Unfortunately, ELLs have long been under-served in U.S. schools, and policies for the instruction of ELL students are not always guided by evidence [7]. Nevertheless, these findings are relevant for other ed tech tools planning implementation in areas with a high concentration of ELLs.

While positive benefits of using MAP Accelerator at recommended levels were observed in the majority of subgroups

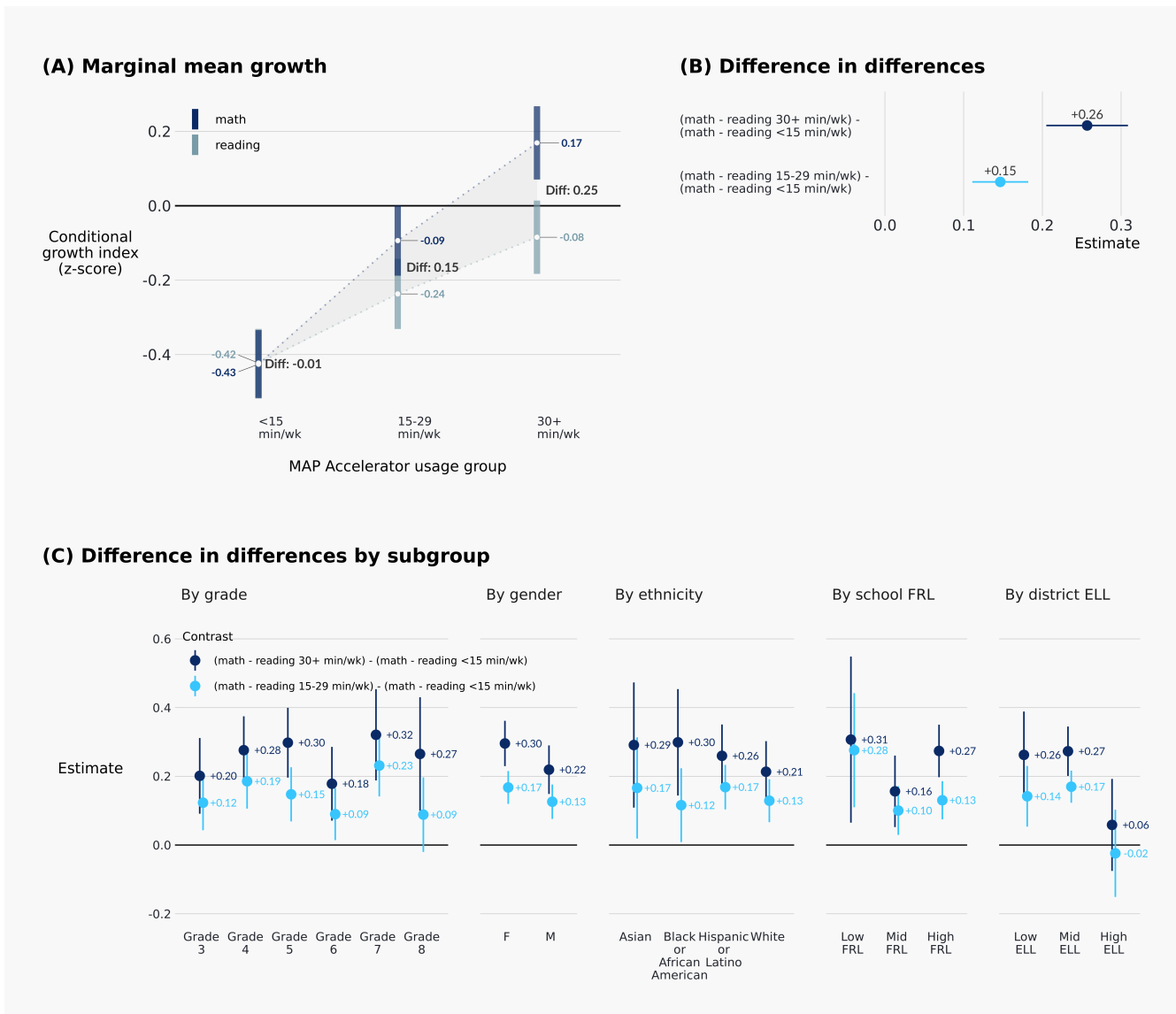


Figure 3: Difference-in-differences effects across subgroups. Error bars are 95% CIs.

that we analyzed, it is important to point out that a low proportion of students actually met this threshold. Exploratory analysis found that usage was mostly predicted by District/School/Teacher factors. This suggests that the degree of class level integration was the largest determinant of whether an individual student would reach recommended usage levels. We can only speculate as to why so many classrooms seemed to not use MAP Accelerator, and the context of the pandemic makes the question even more difficult to address. However, low levels of ed tech usage seems to be a common occurrence [28], suggesting that this challenge is not specific to MAP Accelerator.

4.1 Limitations

There are several limitations of this study that must be addressed. First, although we implemented several statistical and methodological controls, this was still an observational study and does not provide the same quality of causal ev-

idence as a randomized control trial. Our DID approach can only control for unobserved confounds that affect general academic performance, not math specific ones. Second, the study took place during a COVID-disrupted school year. Whether these results will hold post-pandemic is uncertain and yet to be seen. Finally, the results from FRL and ELL analysis leveraged school and district level data, rather than the preferred student level data. Just because a student was in a High ELL group does not mean that they themselves were an ELL.

5. ACKNOWLEDGMENTS

We want to thank Robert Berry, Emma Brunskill, Philip Oreopoulos, and Bi Vuong for their guidance and feedback on this report. We also thank the NWEA research team for their helpful suggestions on an earlier version of our main analysis.

6. REFERENCES

- [1] D. Bates, M. Mächler, B. Bolker, and S. Walker. Fitting linear mixed-effects models using lme4. *Journal of Statistical Software*, 67(1):1–48, 2015.
- [2] T. G. Bond, Z. Yan, and M. Heene. *Applying the Rasch Model: Fundamental Measurement in the Human Sciences*. Routledge/Taylor & Francis Group, New York, NY, 4th edition, 2021.
- [3] L. Boonk, H. J. M. Gijsselaers, H. Ritzen, and S. Brand-Gruwel. A review of the relationship between parental involvement indicators and academic achievement. *Educational Research Review*, 24:10–30, 2018.
- [4] G. Borsato, F. Genesee, K. Lindholm-Leary, B. Saunders, and D. Christian. *Academic Achievement*, page 176–222. Cambridge University Press, 2006.
- [5] L. Cronbach and R. Snow. *Aptitudes and Instructional Methods: A Handbook for Research on Interactions*. New York: Irvington, 1977.
- [6] S. Cunningham. *Causal Inference: The Mixtape*. Yale University Press, 2021.
- [7] Z. de Araujo, S. A. Roberts, C. Willey, and W. Zahner. English learners in k–12 mathematics education: A review of the literature. *Review of Educational Research*, 88(6):879–919, 2018.
- [8] K. A. Ericsson, R. T. Krampe, and C. Tesch-Romer. The role of deliberate practice in the acquisition of expert performance. *Psychological Review*, 100(3):363–406, 1993.
- [9] M. Escueta, V. Quan, A. J. Nickow, and P. Oreopoulos. Education technology: An evidence-based review. Working Paper 23744, National Bureau of Economic Research, 2017. Series: Working Paper Series.
- [10] X. Fan and M. Chen. Parental involvement and students’ academic achievement: A meta-analysis. *Educational Psychology Review*, 13(1):1–22, 2001.
- [11] A. Fredriksson and G. M. d. Oliveira. Impact evaluation using difference-in-differences. *RAUSP Management Journal*, 54(4):519–532, 2019.
- [12] J. Hox. *Multilevel analysis: Techniques and applications*. Routledge/Taylor & Francis Group, 2 edition, 2010.
- [13] V. Irwin, J. Zhang, X. Wang, S. Hein, K. Wang, A. Roberts, C. York, A. Barmer, F. B. Mann, R. Dilig, S. Parker, T. Nachazel, M. Barnett, and S. Purcell. Report on the condition of education 2021. Technical report, U.S. Department of Education, 2021.
- [14] M. Kuhfeld, B. Tarasawa, A. Johnson, E. Ruzek, and K. Lewis. Learning during COVID-19: Initial findings on students’ reading and math achievement and growth. Technical report, NWEA Research, 2020.
- [15] C.-L. C. Kulik, J. A. Kulik, and R. L. Bangert-Drowns. Effectiveness of mastery learning programs: A meta-analysis. *Review of Educational Research*, 60(2):265–299, 1990.
- [16] R. V. Lenth. *emmeans: Estimated Marginal Means, aka Least-Squares Means*, 2021. R package version 1.7.0.
- [17] K. Lewis, M. Kuhfeld, E. Ruzek, and A. McEachin. Learning during COVID-19: Reading and math achievement in the 2020-21 school year. page 12, 2021.
- [18] NWEA. MAP growth technical report. Technical report, NWEA, 2019.
- [19] E. A. Patall, H. Cooper, and J. C. Robinson. Parent involvement in homework: A research synthesis. *Review of Educational Research*, 78(4):1039–1101, 2008.
- [20] S. Roussel, D. Joulia, A. Tricot, and J. Sweller. Learning subject content through a foreign language should not ignore human cognitive architecture: A cognitive load theory approach. *Learning and Instruction*, 52:69–79, 2017.
- [21] V. J. Shute. Focus on formative feedback. *Review of Educational Research*, 78(1):153–189, 2008.
- [22] S. Steenbergen-Hu and H. Cooper. A meta-analysis of the effectiveness of intelligent tutoring systems on college students’ academic learning. *Journal of Educational Psychology*, 106(2):331–347, 2014. ISBN: 1939-2176(Electronic);0022-0663(Print).
- [23] Y. M. Thum and M. Kuhfeld. NWEA 2020 MAP growth achievement status and growth norms for students and schools. Technical report, NWEA Research Report. Portland, OR: NWEA, 2020.
- [24] U.S. Department of Education and Institute of Education Sciences and National Center for Education Statistics. Back-to-school statistics. <https://nces.ed.gov/FastFacts/index.asp>, Accessed = 2022-01-04.
- [25] J. K. Ware. Property value as a proxy of socioeconomic status in education. *Education and Urban Society*, 51(1):99–119, 2019.
- [26] What Works Clearinghouse. Reviews of individual studies, 2020. <https://ies.ed.gov/ncee/wwc/ReviewedStudies>, Accessed: 2022-03-08.
- [27] What Works Clearinghouse. What works clearinghouse standards handbook, version 4.1. Technical report, U.S. Department of Education, Institute of Educational Sciences, National Center for Educational Evaluation and Regional Assistance, 2020.
- [28] C. Wood. Most educational software licenses go unused in k-12 districts, 2018. <https://edscoop.com/most-educational-software-licenses-arent-used-in-k-12-districts/>, Accessed: 2022-03-09.

APPENDIX
A. SUMMARY OF ANALYTIC SAMPLE

	MAP Accelerator usage group				All
	No use	<15 min/wk	15-29 min/wk	30+ min/wk	
<i>Students</i>					
n	74,328	82,393	16,051	8,587	181,359
prct	41%	45%	9%	5%	100%
<i>Locations</i>					
Districts	95	98	91	80	99
Schools	632	563	472	404	649
Teachers	4,334	4,382	2,545	1,625	5,839
Math Classes	7,115	7,405	3,689	2,104	10,464
<i>Gender</i>					
Male	52%	51%	48%	47%	51%
Female	48%	49%	52%	53%	49%
<i>Ethnicity</i>					
Am. Indian / Alaska Native	< 1%	< 1%	< 1%	< 1%	< 1%
Asian	4%	5%	5%	6%	5%
Black or African American	12%	12%	10%	8%	12%
Hispanic or Latino	33%	37%	35%	43%	35%
Multi-ethnic	4%	5%	5%	4%	5%
Native Hawaiian / Pac. Islander	< 1%	1%	< 1%	< 1%	< 1%
White	36%	33%	34%	29%	34%
Not specified / Other	10%	7%	10%	10%	9%
<i>Grade</i>					
Grade 3	19%	15%	19%	21%	17%
Grade 4	19%	17%	17%	21%	18%
Grade 5	18%	19%	17%	20%	18%
Grade 6	13%	18%	21%	19%	17%
Grade 7	15%	18%	14%	11%	16%
Grade 8	16%	14%	12%	8%	14%
<i>School FRL level</i>					
High FRL school	47%	57%	51%	56%	52%
Mid FRL school	25%	24%	31%	26%	25%
Low FRL school	11%	7%	4%	3%	8%
Unreported FRL level	17%	13%	14%	15%	15%
<i>District ELL level</i>					
High ELL district	4%	7%	11%	24%	7%
Mid ELL district	67%	70%	62%	49%	67%
Low ELL district	24%	18%	21%	19%	21%
Unreported ELL level	6%	5%	6%	8%	5%

The Third Workshop of The Learner Data Institute: Big Data, Research Challenges, & Science Convergence in Educational Data Science

Vasile Rus
University of Memphis
vrus@memphis.edu

Stephen E. Fancsali
Carnegie Learning, Inc.
sfancsali@carnegielearning.com

1. WORKSHOP SUMMARY

The Third Workshop of the Learner Data Institute (LDI) builds on the success of two previous, virtual workshops (at EDM 2020 & EDM 2021) and seeks to bring together researchers working across disciplines on data-intensive research of interest to the educational data science and educational data mining communities. In addition to welcoming work describing mature, data-intensive or “big data” research and emerging work-in-progress that spans traditional academic disciplines, the workshop organizers welcome case studies of interdisciplinary research programs and projects, including case studies of learning engineering efforts pursued by universities, learning technology providers, and others (both successful and unsuccessful), as well as position papers on important challenges for researchers harnessing “big data” and crossing disciplinary boundaries as they do so.

We convene researchers and developers from diverse fields who seek to “harness the data revolution” in educational data science and “grow convergence research,” aligning with (at least) two of the U.S. National Science Foundation’s “10 Big Ideas” for emerging research and development opportunities. “Convergence builds and supports creative partnerships and the creative thinking needed to address complex problems” [1], and we expect that bringing together highly experienced researchers, as well as students and early-career researchers, will stimulate substantial growth and interest in state-of-the-art, data-intensive, transdisciplinary or “convergent” approaches to solving vexing societal problems related to education. We also seek to explore the big data and learning engineering frameworks that will enable convergent solutions.

2. WORKSHOP AGENDA

The half-day workshop will begin with an introductory talk presenting a summary report of the work of the LDI and situating its progress within the goals of LDI and the broader notion of convergence research for educational data science. Next, Richard Baraniuk (C. Sidney Burrus Professor of Electrical and Computer Engineering, Rice University & Founder and Director, OpenStax) will deliver a keynote talk (including a question-and-answer period and time for discussion). Workshop organizers aim to include four

to six peer-reviewed contributed research papers (“short” 4-6 page papers, as submitted and informed by peer-review decisions) generally concerned with state-of-the-art big data methodology, applications, and research in educational data science and learning engineering, ideally with an emphasis on science convergence, and 1-3 shorter position papers on similar topics, with an eye toward where future research *should* be directed and/or laying out compelling *challenges* for these areas of research. Each contributed paper presentations will be followed by a question-and-answer session. If there is time, a panel discussion will afford the opportunity for keynote speakers and invited guests to interact with workshop attendees, addressing issues related to convergence research and the future of big data in educational research.

3. WORKSHOP ORGANIZERS

- Vasile Rus, University of Memphis (Co-Chair)
- Stephen E. Fancsali, Carnegie Learning, Inc. (Co-Chair)
- Dale Bowman, University of Memphis
- Jody Cockroft, University of Memphis
- Art Graesser, University of Memphis
- Andrew Hampton, Christian Brothers University
- Philip I. Pavlik Jr., University of Memphis
- Steven Ritter, Carnegie Learning, Inc.
- Deepak Venugopal, University of Memphis

4. ACKNOWLEDGMENTS

The Learner Data Institute is supported by the U.S. National Science Foundation under DRK-12/DIRSE Award #1934745. All opinions and findings stated or implied are solely those of the authors.

5. REFERENCES

- [1] U.S. National Science Foundation. 2017. NSF’s 10 big ideas. https://www.nsf.gov/news/special_reports/big_ideas/index.jsp Last accessed 9 January 2020.

V. Rus and S. Fancsali. The Third Workshop of the Learner Data Institute: Big data, research challenges, & science convergence in educational data science. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 847–847, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license. <https://doi.org/10.5281/zenodo.6853159>

FATED 2022: Fairness, Accountability, and Transparency in Educational Data

Collin Lynch
North Carolina State University
cflynch@ncsu.edu

Anna N. Rafferty
Carleton College
arafferty@carleton.edu

Mirko Marras
University of Cagliari
mirko.marras@acm.org

Steve Ritter
Carnegie Learning
sritter@carnegielearning.com

Mykola Pechenizkiy
Eindhoven University of Technology
m.pechenizkiy@tue.nl

Vinitra Swamy
EPFL
vinitra.swamy@epfl.ch

Renzhe Yu
University of California, Irvine
renzhey@uci.edu

ABSTRACT

The increasing impact of machine learning and algorithmic decision making on education has brought about growing opportunities and concerns. Evidence has shown that these technologies can perpetuate and even magnify existing educational and social inequities. Research on fair machine learning has aimed to develop algorithms that can detect and, in some cases correct, bias, but this effort within the educational data mining community is still limited.

FATED 2022 hopes to spur discussion around algorithmic fairness and bias detection as specifically applied in an educational context. Submissions and panels will be invited to discuss: (a) collection and preparation of benchmark datasets for bias detection and correction tasks, (b) evaluation protocol definition and metric formulation appropriate for bias and fairness in educational tasks, and (c) countermeasure design and development for biased and unfair circumstances. These specific topics will be complemented by a more general discussion of the education-specific challenges for fair machine learning in education, bringing together perspectives from both industry and academia. This workshop builds on the FATED workshop held at EDM 2020, and we expect the workshop to make connections among already interested researchers and provide a foundation for those who want to engage in this area.

Part of the vision of creating adaptive educational technologies and building machine learning systems for education is reducing inequality (e.g., [2]), and data-driven practices are often viewed as a way to make education more equitable (e.g., [1]). While some interventions have been found to decrease achievement gaps (e.g., [4]), there is increasing concern that these systems may instead increase achieve-

ment gaps and perpetuate existing inequities [10, 12]. For example, such systems might make targeted support more available only to students with greater access to technology, or be associated with lower learning gains in more disadvantaged schools (as seen in [11]).

In this workshop, we hope to bring an education-specific lens on broader questions related to fair ML by spurring discussion around:

- **Data Set Collection and Preparation.** By spurring discussion about what educational datasets are particularly ripe for use as benchmarks for detecting and/or correcting bias and what characteristics of an educational dataset make it most useful for measuring or detecting algorithmic bias, this workshop aims to increase awareness about what datasets are available and encourage future research to include results on benchmark datasets.
- **Evaluation Protocol and Metric Formulation.** This workshop encourages discussion about what evaluation protocols and metrics are most suitable for empirical research on fairness and bias across common types of educational machine learning and EDM tasks.
- **Detection and Countermeasure Design.** FATED 2022 provides a forum for discussion about what features of the questions that we address in educational machine learning and the datasets that we use pose particular challenges for detecting and/or addressing algorithmic bias. Further, the workshop will provide an opportunity for researchers to share their work on algorithmic bias detection and correction specifically in education-related context.

Around these themes, FATED 2022 will showcase papers that focus on datasets, evaluation protocol, research, reproducibility, and recently published work (encore papers). By stimulating these discussions, the organizers hope to build community among researchers in this area, including interested EDM researchers who are not yet involved in these topics and fair ML researchers who may wish to engage with the field of education. Surrounding literature from the workshop organizers focuses on educational technology [3, 20, 6, 13, 15], student behavioral patterns [7, 8], algorithmic fairness [19, 18, 5], explainability [17], and responsible analytics for social good [14, 9, 16].

C. Lynch, M. Marras, M. Pechenizkiy, A. Rafferty, S. Ritter, V. Swamy, and R. Yu. FATED 2022: Fairness, accountability, and transparency in educational data. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 848–849, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853079>

1. REFERENCES

- [1] Bill and Melinda Gates Foundation. Ensuring all students receive a public education that equips them to succeed. <https://usprogram.gatesfoundation.org/What-We-Do/K-12-Education>. Accessed: 2022-02-25.
- [2] B. du Boulay, A. Poulouvasillis, W. Holmes, and M. Mavrikis. Artificial intelligence and big data technologies to close the achievement gap. In R. Luckin, editor, *Enhancing Learning and Teaching with Technology*, page 256–285. UCL Institute of Education Press, 2018.
- [3] N. Gitinabard, R. Okoilu, Y. Xu, S. Heckman, T. Barnes, and C. Lynch. Student teamwork on programming projects what can github logs show us? In *Proceedings of The 13th International Conference on Educational Data Mining (EDM 2020)*, pages 409–416, 2020.
- [4] X. Huang, S. D. Craig, J. Xie, A. Graesser, and X. Hu. Intelligent tutoring systems work as a math gap reducer in 6th grade after-school program. *Learning and Individual Differences*, 47:258–265, 2016.
- [5] C. Kung and R. Yu. Interpretable Models Do Not Compromise Accuracy or Fairness in Predicting College Success. In *Proceedings of the 7th ACM Conference on Learning @ Scale (L@S '20)*, pages 413–416, New York, NY, USA, aug 2020. Association for Computing Machinery (ACM).
- [6] Z. Li, L. Yee, N. Sauerberg, I. Sakson, J. J. Williams, and A. N. Rafferty. Getting too personal(ized): the importance of feature choice in online adaptive algorithms. In *Proceedings of the 13th International Conference on Educational Data Mining, EDM 2020, Fully virtual conference, July 10-13, 2020*. International Educational Data Mining Society, 2020.
- [7] Z. Liu, R. Brown, C. F. Lynch, T. Barnes, R. Baker, Y. Bergner, and D. McNamara. Mooc learner behaviors by country and culture; an exploratory analysis. 2016.
- [8] C. F. Lynch. Who prophets from big data in education? new insights and new challenges. *Theory and Research in Education*, 15(3):249–271, 2017.
- [9] M. Mansoury, H. Abdollahpouri, M. Pechenizkiy, B. Mobasher, and R. Burke. Feedback loop and bias amplification in recommender systems. In *Proceedings of the 29th ACM international conference on information & knowledge management*, pages 2145–2148, 2020.
- [10] T. G. Mathewson. Personalized learning can be a tool for equity or a barrier to it. <https://hechingerreport.org/personalized-learning-can-be-a-tool-for-equity-or-a-barrier-to-it/>. The Hechinger Report; Accessed: 2022-02-25.
- [11] M. Meeter. Primary school mathematics during the covid-19 pandemic: No evidence of learning gaps in adaptive practicing results. *Trends in neuroscience and education*, 25:100163, 2021.
- [12] A. Perry and N. Turner-Lee. Ai can disrupt racial inequity in schools, or make it much worse. <https://hechingerreport.org/ai-can-disrupt-racial-inequity-in-schools-or-make-it-much-worse/>. The Hechinger Report; Accessed: 2022-02-25.
- [13] S. Ritter, M. Yudelson, S. E. Fancsali, and S. R. Berman. How mastery learning works at scale. In *Proceedings of the Third (2016) ACM Conference on Learning@ Scale*, pages 71–79, 2016.
- [14] A. Saxena, G. Fletcher, and M. Pechenizkiy. Hm-eiict: Fairness-aware link prediction in complex networks using community information. *Journal of Combinatorial Optimization*, pages 1–18, 2021.
- [15] V. Swamy. Pedagogy, infrastructure, and analytics for data science education at scale, 2018.
- [16] V. Swamy, E. Chen, A. Vankayalapati, A. Aggarwal, C. Liu, V. Mandava, and S. Johnson. Machine learning for humanitarian data: Tag prediction using the HXL standard. 2019.
- [17] V. Swamy, A. Romanou, and M. Jaggi. Interpreting language models through knowledge graph extraction. *arXiv preprint arXiv:2111.08546*, 2021.
- [18] R. Yu, H. Lee, and R. F. Kizilcec. Should College Dropout Prediction Models Include Protected Attributes? In *Proceedings of the Eighth ACM Conference on Learning @ Scale*, pages 91–100, New York, NY, USA, jun 2021. ACM.
- [19] R. Yu, Q. Li, C. Fischer, S. Doroudi, and D. Xu. Towards accurate and fair prediction of college success: Evaluating different sources of student data. In *Proceedings of the 13th International Conference on Educational Data Mining, EDM 2020, Fully virtual conference, July 10-13, 2020*. ERIC, 2020.
- [20] Z. Zakaria, J. Vandenberg, J. Tsan, D. C. Boulden, C. F. Lynch, K. E. Boyer, and E. N. Wiebe. Two-computer pair programming: Exploring a feedback intervention to improve collaborative talk in elementary students. *Computer Science Education*, pages 1–28, 2021.

6th Educational Data Mining in Computer Science Education (CSEDM) Workshop

Bitra Akram
NC State University
bakram@ncsu.edu

Thomas Price
NC State University
twprice@ncsu.edu

Yang Shi
NC State University
yshi26@ncsu.edu

Peter Brusilovsky
University of Pittsburgh
peterb@pitt.edu

Sharon Hsiao
Santa Clara University
ihsiao@scu.edu

ABSTRACT

Computing is an increasingly fundamental skill for students across disciplines. It enables them to solve complex, real and challenging problems and make a positive impact in the world. Yet, the field of computing education is still facing a range of problems from high failure and attrition rates, to challenges in training and recruiting teachers, to the under-representation of women and students of color.

Advanced learning technologies, which use data and AI to improve student learning outcomes, have the potential to address these problems. However, the domain of CS education presents novel challenges for applying these techniques. CS presents domain-specific challenges, such as helping students effectively use tools like compilers and debuggers, and supporting complex, open-ended problems with many possible solutions. CS also presents unique opportunities for developing learning technologies, such as abundant and rich log data, including code traces that capture each detail of how students' solutions evolved over time.

These domain-specific challenges and opportunities suggest the need for a specialized community of researchers, working at the intersection of AI, data-mining and computing education research. The goal of this 6th Educational Data Mining for Computer Science Education (CSEDM) is to bring this community together to share insights for how to understand and support learning in the domain of CS utilizing CS educational data and AI. This field is nascent but

growing, with researching in computing education increasingly using data analysis approaches, and researchers in the EDM community increasingly studying CS datasets. The objective of the CSEDM workshop is to facilitate a discussion among this research community, with a focus on how data mining can be uniquely applied in computing education research. Researchers, faculty and students are encouraged to share their AI- and data-driven approaches, methodologies and experiences where data is transforming the way students learn Computer Science (CS) skills.

We invite researchers who are interested in further exploring, contributing, collaborating and developing data- and AI-driven techniques for building educational tools for Computer Science to submit their papers on any of these topics.

The workshop is a half day workshop. It consists of paper presentations, discussions to facilitate collaboration. Interactive sessions include multiple parallel, short presentations, where participants can float around to the presentations they are interested in, similar to a poster session.

Finally, the workshop celebrates the winners of the 2nd CSEDM Data Challenge. The Data Challenge is an IEDMS and SOLAR-sponsored competition in which researchers compete to develop the best model for a student modeling problem with a CS dataset, including snapshots of student code. Winners are invited to give presentations on their models, followed by discussion of where the challenge focuses in subsequent years.

B. Akram, T. Price, Y. Shi, P. Brusilovsky, and S. I-Han. 6th Educational Data Mining in Computer Science Education (CSEDM) Workshop. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 850–850, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853095>

Rethinking Accessibility: Applications in Educational Data Mining (Virtual only)

Juanita Hicks
American Institutes for Research
jhicks@air.org

Ruhan Circi
American Institutes for Research
rcirci@air.org

Burhan Ogut
American Institutes for Research
bogut@air.org

Darrick Yee
American Institutes for Research
dyee@air.org

Michelle Yin
Northwestern University
michelle.yin@northwestern.edu

ABSTRACT

In standardized testing, the assumption is that assessing all students on the same content under standardized conditions would provide evidence of what students learned and were able to do. However, some facets of standardized testing pose major challenges for students with disabilities (SWDs). We would not expect students with vision impairment, for example, to complete a paper-and-pencil test or a student with learning disabilities to complete a test within the same time limit as students without such disabilities. Thus, to truly include all students, assessment barriers that prevent SWDs, English language learners (ELLs), or other non-traditional students from achieving their best must be removed.

In recent years, technological advances have made computer-based and digitally based assessments (DBAs) more mainstream, especially for large-scale evaluations like the National Assessment of Educational Progress (NAEP). These new DBAs implement technologies to improve accessibility for all participants with different learning backgrounds, especially those with disabilities, using the universal design (UD) framework. UD rests on the idea that in designing tests the full range of students who will participate must be kept in mind from the beginning. By using UD, assessments are expected to become more accessible to and equitable for all students.

Inclusion, diversity, equity, and accessibility (IDEA) is a broad topic and can tap into various areas. Potential topics we would like to cover in our workshop include:

1. Use of different data sources (e.g., large-scale assessments, massive open online courses (MOOCs), games, collaborative chats) to address specific IDEA questions
2. Use of different features and variables, with a special focus on process data (e.g., performance indicators, tool use, accommodation assignments) common and uncommon within IDEA research
3. Special focus on populations, general students with disabilities (SWDs), students with specific disabilities (e.g., autism), English language learners (ELLs)
4. Identification of students who may benefit from IDEA assistance but did not receive it (e.g., students who may need extra time but were not provided extra time)
5. Dissemination and reporting of IDEA results to public audiences for better transparency and inclusion
6. Application of EDM techniques and methods to explore IDEA data (e.g., propensity score matching, machine learning, process mining).

What we hope to accomplish in this workshop is a better understanding of the ways in which research on inclusion, diversity, equity, and accessibility benefit from recent advances and improvements in education and policy (e.g., consistent accommodation procedures), education technology (e.g., digital tests), and emerging research methodologies. Through this workshop we plan to highlight and discover the various approaches and methods through which IDEA research is conducted. Specifically, we aim to explore ways to support this wide research area through data-driven findings.

J. Hicks, R. Circi, B. Ogut, M. Yin, and D. Yee. Rethinking accessibility: Applications in educational data mining. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 851–851, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853065>

Causal Inference in Educational Data Mining

Third Annual Half-Day Workshop

Adam C. Sales, Neil T. Heffernan
Worcester Polytechnic Institute
asales@wpi.edu
nth@wpi.edu

ABSTRACT

The goal in crafting intelligent tutoring systems, educational games, MOOCs, and other computerized learning tools, is to improve student learning. To that end, EDM research typically focuses on methods to identify, measure, and predict learner behaviors or outcomes. Causal research seeks to estimate the impacts of different factors on these behaviors or outcomes—not only predicting who will wheel-spin, experience frustration, or successfully learn a new skill, say, but determining causes these? Causality lies at the heart of both learning science, which seeks to understand how inputs in an educational system affect the system's outputs, and of policy, which seeks to design educational systems that improve learning.

The field of causal inference, which spans statistics, philosophy, economics, computer science, and other more traditional academic disciplines, has itself experienced rapid and exciting developments in the recent past. The new science of causality encompasses new ways of estimating effects under challenging circumstances, such as possible confounding, but also new questions—how do impacts vary between learners? What mechanisms drive causal effects? How may we construct optimal individualized policies for specific learners?

This workshop is intended to raise awareness of the ubiquity and importance of causal questions in EDM, some of the exciting methods available to address those questions, and some of the open questions of causal inference in EDM. It will include invited discussions of ongoing projects addressing causal questions, and short talks about relevant work in progress, including work in any stage of development.

Lastly, the workshop will give an opportunity for EDM researchers to submit open problem related to causality in EDM research. In five minute presentations, researchers will briefly present problems they have encountered in research, or that they just think are interesting, but that they do not yet know how to solve. Each presentation will be followed by an open-ended discussion among the workshop participants, hopefully suggesting ways to solve, or at least better refine the problem. This sub-workshop will hopefully give the presenting researchers constructive suggestions, and spur collaborations. In general, the workshop will be organized to stimulate discussion among participants, including, hopefully, constructive suggestions for open problems.

This workshop will be the third annual EDM workshop of its type—now jointly hosted along with International Conference on Artificial Intelligence in Education. It will follow a similar structure to the previous two workshops, but with all new material.

A. Sales and N. Heffernan. Causal inference in educational data mining. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 852–852, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6853045>

Tutorial: Using the Open Science Framework to promote Open Science in Education Research

Stacy T. Shaw, Adam Sales
Learning Sciences and Technologies
Worcester Polytechnic Institute
sshaw@wpi.edu, asales@wpi.edu

ABSTRACT

Within the past 10 years there has been increasing momentum of the open science movement to make research more open, transparent, and reproducible. However, the adoption of open science practices in education lags behind other fields. In this hybrid tutorial, we will begin by providing a brief overview of open science practices, benefits and workarounds, as well as how the statistical foundations of open science, including the benefits for inference and hypothesis testing. In the second part of the workshop, we will provide a hands-on tutorial of how to use the Open Science Framework to make projects, invite collaborators, preregister studies, share data, code, and materials. Participants in this workshop will gain a better understanding of open science practices, the reasons motivating their adoption, and how to use the Open Science Framework to make their research more open.

Keywords

open science, open access, preprints, preregistration

1. BACKGROUND

The adoption of open science practices— such as preregistration, open data, open code and materials, preprints, and open access— has become increasingly normative across various scientific disciplines, yet attitudes and adoption of these practices remain lagging in education science [1]. While some sub disciplines of education have begun to pioneer the open science movement in education (e.g. special education research, [2]; gifted education, [3]) remaining subfields, such as those related to educational data mining, have not seen widespread adoption. There are many reasons that education scientists may not have adopted open science practices, whether from a lack of training, to general concerns about what can and cannot be shared [4]. Despite the reason, there is a greater need for education about what open science is, how it can be leveraged, as well as guidance on how to use existing resources to make education research more open and transparent.

In this tutorial, we will cover the basics of open science practices, discuss the specific challenges associated with education research (e.g. protecting participant anonymity), and provide a step-by-step tutorial of how to use the Open Science Framework to preregister a study, post open data, code, and materials, and post preprints. We will also discuss further open science activities such as registered reports that might be relevant for participants, and how to use other

S. Shaw and A. Sales. Using the open science framework to promote open science in education research. In A. Mitrovic and N. Bosch, editors, *Proceedings of the 15th International Conference on Educational Data Mining*, pages 853–853, Durham, United Kingdom, July 2022. International Educational Data Mining Society.

© 2022 Copyright is held by the author(s). This work is distributed under the Creative Commons Attribution NonCommercial NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
<https://doi.org/10.5281/zenodo.6852996>

tools to identify opportunities to publish open access (e.g. directory of open access journals, sherpa romeo, etc.).

2. FORMAT AND TIMELINE OF TUTORIAL

The proposed tutorial will be offered in a hybrid fashion, and focuses on open science in education research, and using the Open Science Framework to preregister studies, share data and code, and post preprints. An outline of this workshop can be found below.

- First, we will provide a brief overview of some of the problems of conducting science and research, how open science practices are being used to overcome some of these issues, as well as the myths and workarounds of these practices.
- Second, we will discuss the statistical foundations of open science, including the benefits for inference and hypothesis testing. This discussion will include information about which aspects of data analysis may themselves depend on the data—and hence do not need to be pre-registered, and which phases may not.
- For the third part of this tutorial, we will lead a hands-on tutorial on how to navigate the OSF website, make an account, create projects, invite contributors to join projects, and how to post projects for the public. Afterwards, the presenter will give a step-by-step guide on how to create a preregistration, discuss best steps for preregistration, and identify how to create an embargo on a given preregistration. Other features, such as how to create anonymous versions of projects for blind peer-review will be shown.

3. REFERENCES

- [1] Nosek, B. (2022). Making the Most of the Unconference. Presented at the Unconference on Open Scholarship Practices in Education Research. Available at <https://osf.io/9k6pd/>
- [2] Makel, M. C., Smith, K. N., McBee, M. T., Peters, S. J., & Miller, E. M. (2019). A path to greater credibility: Large-scale collaborative education research. *AERA Open*, 5(4). <https://doi.org/10.1177/2332858419891963>
- [3] Cook, B. G., Lloyd, J. W., Mellor, D., Nosek, B. A., & Therrien, W. J. (2018). Promoting open science to increase the trustworthiness of evidence in special education. *Exceptional Children*, 85(1), 104–118. <https://doi.org/10.1177/0741932516637198>
- [4] Logan, J. A., Hart, S. A., & Schatschneider, C. (2021). Data sharing in education science. *AERA Open*, 7, 23328584211